**МІНІСТЕРСТВО ТРАНСПОРТУ І ЗВ'ЯЗКУ УКРАЇНИ**
**КИЇВСЬКИЙ УІВЕРСИТЕТ ЕКОНОМІКИ І ТЕХНОЛОГІЙ ТРАНСПОРТУ**

# *КОМП'ЮТЕРНІ МЕРЕЖІ*

# *ТА*

# *ТЕЛЕКОМУНІКАЦІЇ*

**Київ – 2006**

**Комп'ютерні мережі та телекомунікації** /Укладачі: Л. Ф. Мараховський, І. О .Марушко— К.: КУЕТТ, 2005. — 333 с.

Укладачі:

**Л. Ф. Мараховський**,

доктор технічних наук, професор кафедри інформаційних систем і технологій Київського університету економіки і технологій транспорту

**І. О .Марушко**,

доцент кафедри інформаційних систем і технологій Київського університету економіки і технологій транспорту

Рецензенти:

**О. І. Стасюк**

доктор економічних наук, професор, завідувач кафедри інформаційних систем і технологій Киіївського університету економіки і технологій транспорту

**В. А. Фабрічев**

доктор технічних наук, професор, завідувач кафедри Національного авіаційного університету

**А. Ю. Рисцова**

Кандидат фізико-математичних наук, доцент кафедри інформаційних систем і технологій Київськьгь університету економіки і технологій транспорту.

# ЗМІСТ

# РОЗДІЛ 1. ОСНОВИ КОМП'ЮТЕРНИХ МЕРЕЖ

## 1. Комп'ютерні мережі

### 1.1. Призначення комп'ютерних мереж

При фізичному з'єднанні двох або більш персональних комп'ютерів (ПК) створюється *комп'ютерна мережа.* Для створення прямого з'єднання ПК, що працюють в операційній системі Windows, є стандартні порти введення/виведення, які використовуються як апаратне забезпечення мережі, та стандартні засоби операційної системи Windows (*Пуск/ Программы/ Стандартные/ Связь/ Прямое кабельное соединение)*, які використовуються як програмне забезпечення мережі.

В загальному вигляді, для створення комп'ютерних мереж потрібно спеціальне апаратне забезпечення (мережні пристрої) та спеціальне програмне забезпечення (мережні програмні засоби).

*Розподілені системи* — це мережі, головна ознака яких — наявність кількох центрів оброблення даних, що дає змогу виконувати паралельні обчислення. До розподілених систем належать мультипроцесорні комп'ютери, багатомашинні системи, комп'ютерні мережі.

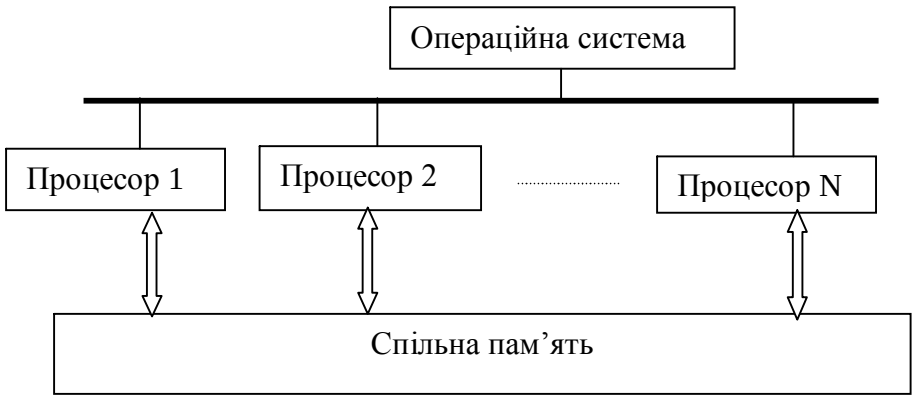



**Рис. 1.1.** Узагальнена структура мультипроцесорного комп'ютера

*Мультипроцесорний комп'ютер* — це комп'ютер з кількома процесорами, кожен з яких незалежно від інших виконує свою програму обчислення, а

взаємообмін даними між ними відбувається через спільну оперативну пам'ять (рис.1.1.).

Вперше цю модель, в якій усі процесори мають спільну операційну систему (ОС), що керує обчислювальним навантаженням між процесорами, розглядали ще в першій половині 60-х років в СРСР. Найголовніша особливість — це висока продуктивність при можливості паралельної роботи в обчислювальному процесі. Але при лінійному обчислювальному процесі — це великі апаратурні витрати на одну операцію.

*Багатомашинна система* — це обчислювальний комплекс, що містить кілька комп'ютерів, кожен з яких працює під управлінням власної ОС (рис.1.2.).

Вперше цю модель, в якій апаратні і програмні зв'язки між комп'ютерами здійснювала інша ЕОМ, запропонували в 60-х роках в США. Зв'язки між комп'ютерами (ЕОМ) забезпечують роботу всіх комп'ютерів як єдиного цілого. Програмні та апаратні засоби зв'язків мають високу швидкість і містять програмні модулі, що розподіляють обчислювальне навантаження та

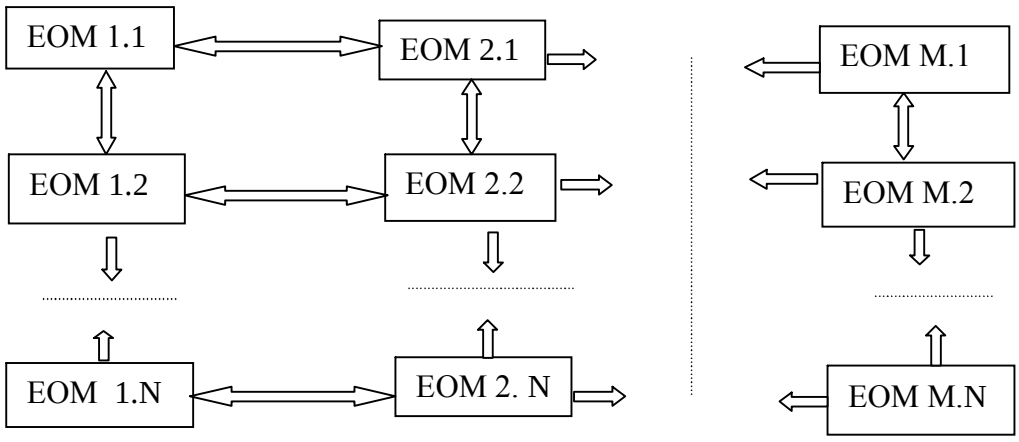



**Рис. 1.2.** Узагальнена структура багатомашинної системи

здійснюють синхронізацію обчислень. У багатомашинних системах, як і у мультипроцесорних, досить висока продуктивність та висока відмовостійкість.

*Комп'ютерна мережа* — сукупність взаємопов'язаних (через канали передачі даних) комп'ютерів, які забезпечують користувачів засобами обміну інформацією і колективного використання ресурсів мережі: апаратних, програмних та інформаційних.

Всі комп'ютерні системи мають одне призначення — забезпечення сумісного доступу до спільних ресурсів. Розрізняють ресурси трьох типів: *апаратні, програмні* і *інформаційні.* Прикладом апаратних ресурсів мережі може бути пристрій друку (принтер), який є спільним для локальної комп'ютерної мережі. Прикладом програмних ресурсів мережі може бути застосування можливостей програмного забезпечення віддаленої ЕОМ для розв'язання задачі і отримання від неї результатів на свій ПК. Прикладом інформаційного ресурсу можуть бути дані, які зберігаються на віддаленій ЕОМ.

При практичній роботі в комп'ютерній мережі будь-якого типу одночасно використовуються всі три типи ресурсів. Прикладом цього може бути робота в мережі Internet: ми використовуємо чужі апаратні засоби, на яких працюють чужі програми, які використовують потрібні нам дані.

З розвитком мереж, створюваних різними розробниками з використанням різних типів комп'ютерів, постала проблема стандартизації в мережах, яка б забезпечила можливість сумісного функціонування різних мереж.Питаннями стандартизації електронних пристроїв традиційно займаються кілька установ в США. Це Інститут інженерів електриків і елетроніки (IEEE), асоціація EIA (Elektronics Industrie Association), ITU (International Telecommunication Union), міжнародна організація стандартів ISO (International Standard Organistion) і інші. В Європі – це Європейська асоціація виробників комп'ютерів ( ECMA) Розроблювані комп'ютерні мережі і відповідні технології повинні задовольняти відповідним стандартам. В противному разі розроблювані пристрої не матимуть доступу до ринку. Кожна з цих організацій має свій формат розробленого нею стандарту. Наприклад, формат стандатру ISO має вигляд XXXXISO, де XXXX-відповідне чотризначне число. Приклад стандарту IEEE для комп'ютерної мережі Token Ring –802.5. Приклад стандарту ECMA- ECMA-80.

## 1.2. Об'єкти комп'ютерної мережі

Розглянемо кілька загальних визначень, що стосуються інформаційних систем:

- *Реальна система* -це сукупність кількох ЕОМ з відповідним програмним забезпеченням (ПЗ) та периферійним обладнанням, яка повністю автономно передає інформацію. Вона автономна, якщо не приєднана до мережі.
- *Реальна кінцева система* (real end system) –це реальна система, що виконує в мережі функції станції даних, тобто є джерелом або споживачем інформації ( станція-аппаратура, яка виконує функції приймання та передавання інформації).
- *Відкрита система*- це система, яка побудована та функціонує з дотриманням вимог міжнародних стандартів.
- *Комунікаційна система*- це реальна відкрита система, яка забезпечує обмін даними між абонентськими системами у відкритій інформаційній системі.
- *Абонентська система* (АС)-це реальна відкрита система, яка є постачальником або споживачем ресурсів мережі, забезпечує доступ до них користувачів і керує взаємозв'язком відкритих систем (сукупність робочої станції і абонента). Ініціатором та учасником обміну в АС являються прикладні процеси.
- *Прикладний процес* – це процес у реальній кінцевій системі, який опрацьовує інформацію для визначених потреб. Прикладним процесом можуть бути дії оператора за терміналом, програми доступу до бази даних, програма керування технологічним процесом і таке інше. Зв'язок між прикладними процесами реалізується за допомогою середовища передавання даних.
- *Середовище передавання* даних- це сукупність ліній передавання даних та, можливо, іншого обладнання, яке забезпечує передавання даних між станціями. Частина прикладного процесу, яка відповідає за організацію зв'язку називається прикладним об'єктом

*Фізичне передавальне середовище* є основа комунікаційної системи, що забезпечує передачу інформації між абонентськими системами (рис.1.3.).

.


**Рис. 1.3.** Фізичне передавальне середовище

Між прикладними об’єктами в мережі є сполучення, що пролягає через середовище зв’язку відкритих систем.

- Середовище зв’язку відкритих систем – це сукупність функцій, які дають змогу реальним відкритим системам виконувати обмін даними відповідно до міжнародних стандартів. Структуру середовища зв’язку відкритих систем визначає стандарт 7498 ISO. Середовище має складний набір функцій, що потребує при його створенні дотримання ієрархічного підходу:

    1. Складну функцію передавання розділяють на рівні.
    2. Кожен рівень виконує конкретний скінчений набір завдань.
    3. Межі між рівнями визначають так, щоб необхідний обмін був мінімальним.
    4. Рівні описують так, щоб зміна одного рівня не спричиняла зміни інших.

Реалізується функція передавання сукупністю програмних та апаратних засобів. До апаратних засобів відносяться адаптери.

*Адаптери (мережні адаптери)* — це технічні пристрої, які виконують функції сполучення ЕОМ з каналами зв’язку і розрізняються методами доступу

до середовища та протоколами..

За допомогою зв'язку ПК може дістати доступ до ресурсів іншого ПК. Ця властивість називається *прозорістю мережі.* Комп'ютерні мережі дають такі переваги підприємству, як розподіл ресурсів великої вартості, вдосконалення комунікацій, розширення доступу до інформації і таке інше.

## 1.3. Класифікації комп'ютерних мереж

Комп'ютерні мережі можна класифікувати за різними ознаками. Приведемо класифікації за найбільш характерними ознаками:

- за розмірами території, що охоплює мережа:

- *Глобальні мережі* — об'єднують користувачів, розташованих по всьому світу на відстані 10-15 тис. км. один від одного. В них використовуються супутникові канали зв'язку. Вони можуть бути національними або транснаціональними.

*Регіональні мережі* — об'єднують користувачів міста, області, невеликих країн на відстані 10-1000 км. Канали зв'язку в них застосовуються телефонні лінії. Вони можуть бути міськими або регіональними.

*Локальні обчислювальні мережі* (ЛОМ) —сполучають абонентів одного або кількох сусідніх будівель одного підприємства, установи. Такі мережі також називаються *корпоративними системами.* Для зв'язку між комп'ютерами використовують спеціальні (швидкісні) канали зв'язку. Відстань між ЕОМ до 10-20 км. Розмір мережі може досягати 100 км. Як видно, територіальною ознакою локальні мережі частково перекриваються з регіональними. Тут слід враховувати, що важливою ознакою локальної мережі, на відміну від регіональних, є швидкісний зв'язок в мережі..

- за сферою застосування:

- офісні,
- побутові,

- промислові,

- за архітектурним вирішенням (технологією), що виражається у фірмовій назві:

- Ethernet,
- Token Ring,
- Arcnet,
- FDDI,
- ATM…

- за топологією:

- шинна, коли комп’ютери з’єднані послідовно однією шиною,
- кільцева, коли шина, що з’єднує комп’ютери, замкнена в кільце,
- зіркоподібна, коли всі комп’ютери під’єднані до одного спільного пристрою-концентратора,
- комбінована, що може включати вище названі топології і створювати деревоподібні (ієрархічні) структури,

- за фізичним середовищем передавання:

- мідний провід –коаксіальний кабель товстий та тонкий, скручена пара екранована та неекранована,
- оптоволоконний кабель.
- ефір-для передавання сигналів електромагнітної природи: лазерне, інфрачервоне, мікрохвильове випрмінювання, сигнали низькоорбітальних та високоорбітальних супутників,

- за методом доступу до мережі:

- циклічний,
- маркерний,

- метод опитування,
- конкурентний,
- з запитом пріорітету.

Для забезпечення сумісності апаратних та програмних засобів в комп'ютерних мережах застосовують спеціальні стандарти, які мають назву *протоколів.* Вони визначають характер апаратних взаємодій компонентів мережі (апаратні протоколи) і характер програмних взаємодій і даних (програмні протоколи).Фізичну функцію підтримки протоколів виконують апаратні пристрої (інтерфейси) і програмні засоби (програми підтримки протоколів). Програми, що виконують підтримку протоколів, часто теж називають протоколами.

Групи співробітників, які працюють над одним проектом у рамках локальної мережі, називають *робочими групами.* В одній локальній мережі можуть працювати декілька робочих груп. У співробітників робочих груп можуть бути різні права доступу до загальних ресурсів мережі. Сукупність прийомів розділення й обмеження прав учасників комп'ютерної мережі має назву *політики мережі.* Управління політикою мережі (їх може бути декілька в одній мережі) займається адміністратор мережі (або системний адміністратор).

Для зв'язку між локальними мережами, які працюють за різними протоколами, використовуються інтелектуальні засоби сполучення локальних мереж такі як *мости, комутатори* та *шлюзи* . Шлюзи можуть бути як апаратні, так і програмні. В останньому випадку комп'ютер може виконувати не тільки функції шлюзу, але і інші функції робочої станції.

Важливу роль при підключенні локальної мережі підприємства до глобальної мережі відіграє поняття *безпеки мережі.* Для забезпечення безпеки установлюються *брандмауери* між локальною і глобальною мережами. Брандмауером може бути спеціальний ПК або комп'ютерна програма, що не дозволяє несанкціонованому переміщенню даних між мережами.

Зв’язок між комп’ютерами в мережі і між мережами здійснюється по каналах зв’язку.. Каналом зв’язку називають сукупність програмно-технічних засобів та фізичного середовища передавання, призначених для передавання сигналу даних. Вони стандартизовані за швидкістю передавання і мають свої назви. Наприклад Switshed 56- канал на 56Мбіт/с, Т1-на 1,544Мбіт/с, Т3 –на 45Мбіт/с. Це приклади американських стандартів. Існують європейські стандарти, наприклад, близький до Т1, але не співпадає з ним канал європейського стандарту Е1.

## 1.4. Методи передавання даних у мережах ЕОМ

При обміні даними між вузлами використовуються три методи передавання даних:

- симплексна (односпрямована) передача (телебачення, радіо);
- напівдуплексна ( прийом-передача інформації здійснюється по черзі);
- дуплексна (двунаправлена), кожна станція одночасно передає та приймає дані.

Для передавання даних в інформаційних системах найбільш часто застосовується послідовна передача. Широко використовуються наступні методи послідовної передачі: асинхронна і синхронна.

При асинхронній передачі кожен символ передається окремою посилкою (рис. 1.4).

*При асинхронній передачі* кожен символ передається окремою посилкою. Стартові біти попереджають приймач про початок передачі. Потім передається символ. Для визначення вірогідності передачі використовується біт парності (біт парності =1, якщо кількість одиниць у символі непарна, і 0 — у противному випадку. Останній біт («стоповий біт») сигналізує про закінчення передачі.

*Переваги*: нескладна відпрацьована система; недорогий (у порівнянні із синхронним) інтерфейс устаткування.

*Недоліки*: третя частина пропускної здатності губиться на передачу службових бітів (старт-стопових і біта парності); невисока швидкість передачі в

порівнянні із синхронною; при множинній помилці за допомогою біта парності неможливо визначити вірогідність отриманої інформації.

*Асинхронна передача* використовується в системах, де обмін даними відбувається час від часу і не потрібна висока швидкість передачі даних. Деякі системи використовують біт парності як символьний біт, а контроль інформації виконується на рівні протоколів обміну даними (Xmodem, Zmodem, MNP).

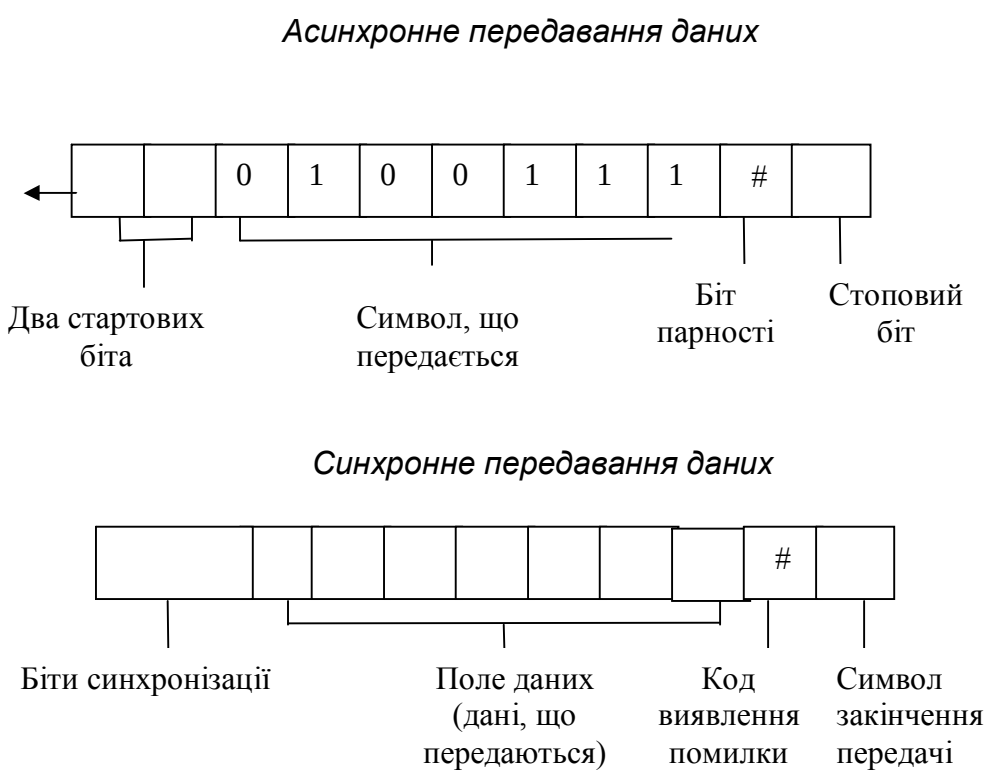



**Рис. 1.4.** Асинхронне і синхронне передавання даних

При використанні синхронного методу дані передаються блоками. Для синхронізації роботи приймача і передавача на початку блоку передаються біти синхронізації. Потім передаються дані, код виявлення помилки і символ закінчення передачі. При синхронній передачі дані можуть передаватися і як символи, і як потік бітів. Як код виявлення помилки звичайно використовується ц*иклічний надлишковий код виявлення помилок* (CRC). Він обчислюється по вмісту подання даних і дозволяє однозначно визначити вірогідність прийнятої інформації.

*Переваги*: висока ефективність передачі даних; високі швидкості передачі даних; надійний вбудований механізм виявлення помилок.

*Недоліки*: інтерфейс устаткування більш складний і, відповідно, більш дорогий.

Протоколи SDLC і HDLC ґрунтуються на синхронній бітово-орієнтованій передачі даних.

## 1.5. Питання для самоконтролю

1. Яким чином створюються комп‘ютерні мережі?
2. Що треба мати для з’єднання ПК в мережу?
3. Що можна віднести до розподілених систем?
4. Що таке мультипроцесорний комп‘ютер?
5. Що таке багатомашинна система?
6. Що таке комп‘ютерна мережа?
7. Яке призначення комп’ютерної системи?
8. Що таке відкрита система?
9. Що таке комунікаційна система?
10. Яку роль відіграють стандарти для мереж?
11. Які існують класифікації мереж?
12. Типи ресурсів.
13. Що таке програмний ресурс?
14. Що таке апаратний ресурс?
15. Що таке інформаційний ресурс?
16. Що таке абонентні системи?
17. Що таке станція?
18. Що забезпечує фізичне передавальне середовище?
19. Що таке прозорість мережі?
20. Які переваги мають комп’ютерні мережі?
21. Класифікація мереж за територіальними ознаками.
22. Глобальна комп’ютерна мережа.
23. Регіональна комп’ютерна мережа.

24. Локальна комп’ютерна мережа.
25. Що визначає протокол? Що таке робоча група?
26. Що таке шлюз?
27. Які бувають шлюзи?
28. Що таке безпека мережі?
29. Що таке брандмауер?
30. Що таке політика мережі?
31. Які функції адміністратора мережі?
32. Що таке канали зв’язку?
33. Які три методи передачі даних використовуються при обміні даними між вузлами?
34. Асинхронне передавання даних. Переваги та недоліки.
35. Синхронне передавання даних. Переваги та недоліки.

# 2. Архітектурні принципи побудови мереж

## 2.1. Модель взаємозв’язку відкритих систем.

Розвиток засобів обчислювальної техніки, а особливо поява персональних комп'ютерів привело до створення нового типу інформаційно-обчислювальних систем за назвою локальна обчислювальна мережа (ЛОМ).

ЛОМ знайшли широке застосування в системах автоматизованого проектування і технологічної підготовки виробництва, системах керування виробництвом і технологічними комплексами, у конторських системах, бортових системах керування і т.д. ЛОМ є ефективним способом побудови складних систем керування різними виробничими підрозділами. ЛОМ інтенсивно впроваджуються в медицину, сільське господарство, освіту, науку й таке ін.

Локальна мережа — (LAN — Local Area Network), дана назва відповідає об'єднанню комп'ютерів, розташованих на порівняно невеликій території (одного підприємства, офісу, однієї кімнати). Існуючі стандарти для ЛОМ

забезпечують зв'язок між комп'ютерами на відстані від 2,5 км до 6 км (Ethernet і ARCNET, відповідно).

ЛОМ — набір апаратних засобів і алгоритмів, що забезпечують з'єднання комп'ютерів, інших периферійних пристроїв (принтерів, дискових контролерів і т.ін.) і дозволяють їм спільно використовувати загальну дискову пам'ять, периферійні пристрої, обмінюватися даними.

Інформаційні системи, у яких засоби передачі даних належать одній компанії і використовуються тільки для цієї компанії, прийнято називати корпоративна мережа підприємства (Enterprise Network). Для автоматизації роботи виробничих підприємств часто використовуються системи на базі протоколів MAP/TOP:

MAP (Manufacturing Automation Protocol) — мережа для виробничих підприємств, заводів (виконується автоматизація роботи конструкторських відділів і виробничих, технологічних цехів). MAP дозволяє створити єдиний технологічний ланцюжок від конструктора, що розробив деталь, до устаткування, на якому виготовляють цю деталь.

TOP (Technical and Office Protocol) — протокол автоматизації технічної й адміністративної установи.

MAP/TOP - системи, що в цілому автоматизують роботу виробничого підприємства.

Основне призначення ЛОМ — у розподілі ресурсів ЕОМ: програм, сумісності периферійних пристроїв, терміналів, пам'яті. Отже, ЛОМ повинна мати надійну і швидку систему передачі даних, вартість якої повинна бути менше в порівнянні з вартістю підключених робочих станцій, що являється головною ознакою системи з розподіленою обробкою даних. Іншими словами, вартість передавання одиниці інформації повинна бути значно нижче вартості обробки інформації в робочих станціях. Виходячи з цього, ЛОМ, як система розподілених ресурсів, повинна ґрунтуватися на наступних принципах:

- єдиного передавального середовища;
- єдиного методу керування;

- єдиних протоколів;
- гнучкої модульної організації;
- інформаційної і програмної сумісності.

В цілому на сьогодні однією з важливих ознак локальної мережі є наявність швидкісних каналів зв’язку, що дозволяє створювати досить великі мережі (до 100 км.), що задовольняють умові розподіленої обробки даних (час передавання менше часу оброблення інформації).

Міжнародна організація по стандартизації (ISO), ґрунтуючись на досвіді багатомашинних систем, що був накопичений у різних країнах, висунула концепцію архітектури відкритих систем — еталонну модель, використовувану при розробці міжнародних стандартів – стандарт 7498 ISO.

На основі цієї моделі обчислювальна мережа являється розподіленим обчислювальним середовищем, що включає в собі велику кількість різноманітних апаратних і програмних засобів. По вертикалі дане середовище пропонує кілька логічних рівнів, на кожний з який покладена одна з задач мережі. По горизонталі інформаційно-обчислювальне середовище поділяється на локальні частини (відкриті системи), що відповідають вимогам і стандартам структури відкритих систем.

Частина відкритої системи, що виконує деяку функцію зв’язку і входить до складу того чи іншого рівня, називається об'єктом.

Правила, за якими здійснюється взаємодія об'єктів одного й того ж рівня, називаються протоколом (протокольне керування).

Передавання між об’єктами суміжних рівнів визначається інтерфесами (інтерфейсне керування).

Протоколи визначають порядок обміну інформацією між мережними об'єктами. Вони дозволяють взаємодіючим робочим станціям посилати один одному виклики, інтерпретувати дані, обробляти помилкові ситуації і виконувати безліч інших різних функцій. Суть протоколів полягає в регламентованих обмінах точно специфікованими командами і відповідями на

них (наприклад, призначення фізичного рівня зв'язку — передача блоків даних між двома пристроями, підключеними до одного фізичного середовища).

Кожен рівень підрозділяється на дві частини:

- специфікацію послуг;
- специфікацію протоколу.

Специфікація послуг визначає, що робить рівень, а специфікація протоколу — як він це робить.

Причому, кожен конкретний рівень може мати більше одного протоколу.

Велике число рівнів, використовуваних у моделі, забезпечує декомпозицію інформаційно-обчислювального процесу на прості складові. У свою чергу, збільшення числа рівнів викликає необхідність включення додаткових зв'язків відповідно до додаткових протоколів і інтерфейсів. Інтерфейси (макрокоманди, програми) залежать від можливостей використовуваної ОС.

Головні функції будь-якого N-го рівня можна описати в загальному вигляді, хоча не всі вони реалізуються на окремих рівнях:

1. Вибір протоколу.
2. Налагодження та розривання сполучень.
3. Мультиплексування або розщеплення. Мультиплексування має місце, коли кільком сполученням верхнього (N+1) рівня відповідає одне сполучення нижнього (N) рівня. Розщеплення має місце, коли одному сполученню верхнього рівня відповідає кілька сполучень нижнього рівня.
4. Передавання даних. Взаємодія реалізується за допомогою протокольних блоків нижнього рівня, які містять протокольну інформацію керування та дані користувача (*протокольний блок*), передані об’єктом верхнього рівня.
5. Керування потоком даних між об’єктами – протокольне та інтерфейсне.
6. Сегментування або зчеплення даних. Якщо інтерфейсний блок вищого рівня більший ніж допустимий розмір протокольного блоку

нижчого рівня, то здійснюється сегментування блоку. Зчеплення - зворотній процес.

7. Організація послідовності передавання.

8. Захист від помилок – на всіх рівнях.

Міжнародна організація по стандартизації запропонувала семирівневу модель, якій відповідає і програмна структура (рис. 2.1.).


**Рис. 2. 1.** Рівні управління і протоколи ЛОМ

Розглянемо функції, що виконуються кожним рівнем програмного забезпечення:

1. Фізичний — здійснює як з'єднання з фізичним каналом, так і розривання, керування каналом, а також визначає швидкість передавання даних і топологію мережі. На фізичному рівні протокольний блок приймає вигляд послідовності бітів.

2. Канальний — здійснює обрамлення переданих масивів інформації допоміжними символами і контроль переданих даних. На цьому рівні протокольний блок називають *кадром*..

3. Мережний — визначає маршрут передачі інформації між мережами (ПЕОМ), забезпечує обробку помилок, а також керування потоками даних.

Основна задача мережного рівня — маршрутизація даних (передача даних між мережами). Спеціальні пристрої — маршрутизатори (Router) визначають, якій мережі призначене те чи інше повідомлення і направляють цю посилку в задану мережу. Для визначення абонента усередині мережі використовується адреса вузла (Node Address). Для визначення шляху передачі даних між мережами на маршрутизаторах будуються таблиці маршрутів (Routing Tables), що містять послідовність передачі даних через маршрутизатори. Кожен маршрут містить адресу кінцевої мережі, адресу наступного маршрутизатора і вартість передачі даних по цьому маршруту. При оцінці вартості можуть враховуватися кількість проміжних маршрутизаторів, час, необхідний для передавання даних, просто грошова вартість передачі даних по лінії зв'язку. Для побудови таблиць маршрутів найбільш часто використовують або метод векторів або статичний метод. При виборі оптимального маршруту застосовують динамічні або статичні методи. На мережевому рівні протокольні блоки називають *пакетами*. Кожен пакет містить адреси джерела і місця призначення, а також засобу виявлення помилок На мережному рівні можливе застосування однієї з двох процедур передачі пакетів:

- датаграм — тобто, коли частини повідомлення (пакети) незалежно доставляється адресату різними маршрутами, обумовленими сформованою динамікою в мережі. При цьому кожен пакет містить у собі повний заголовок з адресою одержувача. Процедури керування передаванням таких пакетів по мережі називаються датаграмною службою;
- віртуальних з'єднань — коли встановлення маршруту передачі всього повідомлення від відправника до одержувача здійснюється за допомогою спеціального службового пакета — запиту на з'єднання. У такому випадку для

цього пакета вибирається маршрут, який, при позитивній відповіді одержувача на з'єднання, закріплюється для всього наступного трафіка (потоку повідомлень у мережі передачі даних), і одержується номер відповідного віртуального каналу (з'єднання) для подальшого використання його іншими пакетами того ж повідомлення. Пакети, що передаються по одному віртуальному каналу, не є незалежними і тому включають скорочений заголовок, що включає порядковий номер пакета, що належить одному повідомленню.

*Недоліки*: значна в порівнянні з датаграмою складність у реалізації, збільшення накладних витрат, викликаних встановленням і роз'єднанням повідомлень.

*Висновок*. Датаграмний режим зручніше використовувати для мереж складної конфігурації, де значне число ЕОМ у мережі, ієрархічна структура мережі, надійність, вірогідність передачі даних по каналах зв'язку, довжина пакета більш 512 байт.

4. Транспортний — зв'язує нижні рівні (фізичний, канальний, мережний) з верхніми рівнями, що реалізуються програмними засобами. Цей рівень немов би відділяє засоби формування даних у мережі від засобів їхнього передавання. Тут здійснюється поділ інформації на порції визначеної довжини й уточнюється адреса призначення. Транспортний рівень дозволяє мультиплексувати передані повідомлення чи з'єднання. Мультиплексування повідомлень дозволяє передавати повідомлення одночасно по декількох лініях зв'язку, а мультиплексування з'єднань — передає в одній посилці кілька повідомлень для різних з'єднань.

5. Сеансовый — на даному рівні здійснюється керування сеансами зв'язку між двома взаємодіючими користувачами (визначає початок і закінчення сеансу зв'язку: нормальне чи аварійне; визначає час, тривалість і режим сеансу зв'язку; визначає точки синхронізації для проміжного контролю і відновлення при передачі даних; відновлює з'єднання після помилок під час сеансу зв'язку без втрати даних.

6. Представницький — керує представленням даних у необхідній для програми користувача формі, генерацією й інтерпретацією взаємодії процесів, кодуванням/декодуванням даних. На робочих станціях можуть використовуватися різні операційні системи: DOS, UNIX, OS/2. Кожна з них має свою файлову систему, свої формати збереження й обробки даних. Задачею даного рівня є перетворення даних при передаванні інформації у формат, що використовується в інформаційній системі. При прийманні даних рівень представлення даних виконує зворотне перетворення. У такий спосіб з'являється можливість організувати обмін даними між станціями, на яких використовуються різні операційні системи.

Формати представлення даних можуть розрізнятися за наступними ознаками:

- порядок проходження бітів і розмір символу в бітах;
- порядок проходження байтів;
- представлення і кодування символів;
- структура і синтаксис файлів.

Кодування переданої інформації забезпечує захист її від перехоплення.

7. Прикладний — у його веденні знаходяться прикладні мережні програми, що обслуговують файли, а також він виконує обчислювальні, інформаційно-пошукові роботи, логічні перетворення інформації, передачу поштових повідомлень і т.ін. Головна задача цього рівня — забезпечити зручний інтерфейс для користувача.

На різних рівнях обмін відбувається різними одиницями інформації: біти, кадри, пакети, сеансові повідомлення, користувацькі повідомлення.

*Протоколи в ЛОМ*

Організація ЛОМ базується на принципі багаторівневого керування процесами, що включають у себе ієрархію протоколів і інтерфейсів.

Протокол УФК визначає форму представлення і порядок передачі даних через фізичний канал зв'язку, фіксує початок і кінець кадру, що несе в собі дані, формує і приймає сигнал зі швидкістю, властивою пропускній здатності каналу.

Канальний рівень можна розділити на два підрівні: керування доступом до каналу (УДК) і керування інформаційним каналом (УІК).

Протокол УДК встановлює порядок передачі даних через канал, вибірку даних.

Протокол УІК забезпечує вірогідність даних, тобто формуються коди перевірки при передаванні даних.

У багатьох ЛОМ відпадає необхідність у мережному рівні. До нього звертаються при з'єднування кількох ЛОМ, що містять моноканали.

Протокол УП забезпечує транспортний інтерфейс, що ліквідує розходження між потребами процесів в обміні даними й обмеженнями інформаційного каналу, який організується нижніми рівнями керування. Протоколи вищих рівнів — УС, УПД, УПП — за своїми функціями аналогічні відповідним протоколам глобальних мереж, тобто реалізується доступ терміналів до процесів, програм, до вилучених файлів, передавання файлів, вилучення уведених завдань, обмін графічною інформацією й ін.

## 2.3. Організація взаємодії пристроїв у мережі.

В залежності від способу організації обробки даних і взаємодії користувачів, що підтримується конкретною мережною операційною системою, виділяють два типи інформаційних систем:

- ієрархічні мережі;
- мережі клієнт-сервер.

*В ієрархічних мережах* усі задачі, зв'язані зі збереженням, обробкою даних, їхнім представленням користувачам, виконує центральний комп'ютер. Користувач взаємодіє з центральним комп'ютером за допомогою термінала. Операціями введення-виведення інформації на екран керує центральний комп'ютер.

*Переваги ієрархічних систем*:

- відпрацьована технологія забезпечення відмовостійкості, схоронності даних;
- надійна система захисту інформації і забезпечення таємності.

*Недоліки:*

- висока вартість апаратного і програмного забезпечення, високі експлуатаційні витрати;
- швидкодія і надійність мережі залежать від центрального комп'ютера.

Приклади ієрархічних систем є SNA, IBM Corp., DNA, DEC.

*У системах клієнт-сервер* обробка даних розділена між двома об'єктами: клієнтом і сервером. Клієнт — це задача, робоча станція, користувач. Він може сформувати запит для сервера: представити файл, здійснити пошук запису і т.ін. Сервер — це комп'ютер, що виконує обробку запиту. Він відповідає за збереження даних, організацію доступу до цих даних і передачу даних клієнту. У системах клієнт-сервер навантаження по обробці даних розподілено між клієнтом і сервером, тому вимоги до продуктивності комп'ютерів, використовуваних як клієнт і сервер, значно нижче, ніж в ієрархічних системах.

За організацією взаємодії прийнято розрізняти два типи систем, що використовують метод клієнт-сервер:

- рівноправна мережа;
- мережа з виділеним сервером.

*Рівноправна мережа (одноранговa)* — це мережа, у якій немає єдиного центра керування взаємодією робочих станцій, немає єдиного пристрою збереження даних. Операційна система такої мережі розподілена по всіх робочих станціях, тому кожна робоча станція одночасно може виконувати функції як сервера, так і клієнта. Користувачу в такій мережі доступні всі пристрої (принтери, тверді диски і т.ін.), підключення до інших робочих станцій.

*Переваги*: низька вартість (використовуються всі комп'ютери, підключені до мережі, й помірні ціни на програмне забезпечення (ПЗ) для роботи мережі);

висока надійність (при виході з ладу однієї робочої станції, доступ припиняється лише до деякої частини інформації).

*Недоліки*: робота мережі ефективна тільки при кількості одночасно працюючих станцій не більш 10; труднощі організації ефективного керування взаємодією робочих станцій і забезпечення таємності інформації; труднощі відновлення і заміни ПЗ робочих станцій.

*Мережа з виділеним сервером* — тут один з комп'ютерів виконує функції збереження даних загального користування, організації взаємодії між робочими станціями, виконання сервісних послуг — сервер мережі. На такому комп'ютері працює операційна система і всі поділювані пристрої (тверді диски, принтери, модеми і т.ін.) під'єднуються до неї, виконується збереження даних, друк завдань, вилучення і обробка завдань. Робочі станції взаємодіють через сервер, тому логічну організацію такої мережі можна представити топологією типу «зірка», де центральний пристрій — це сервер.

*Переваги*: вище швидкість обробки даних (визначається швидкодією центрального комп'ютера, і на сервері встановлюється спеціальна мережна операційна система, яка розрахована на обробку і виконання запитів, що надійшли одночасно від кількох користувачів); має надійну систему захисту інформації і забезпечення таємності; простіше в керуванні в порівнянні з рівноправними.

*Недоліки*: така мережа дорожча через окремий комп'ютер під сервер; менш гнучка в порівнянні з рівноправною.

Мережі з виділеним сервером  більш розповсюджені. Приклади мережних операційних систем такого типу є LAN Server, IBM Corp., VINES, Banyan System Inc., NetWare, Novell Inc.

## 2.2. Питання для самоконтролю

1. Що привело до створення нового типу інформаційно-обчислювальних систем за назвою локальна обчислювальна мережа (ЛОМ)?
2. Де знайшли широке застосування ЛОМ?
3. Що таке ЛОМ?

4. Як на сьогодні прийнято поділяти інформаційно-обчислювальні системи? Що таке LAN (Lokal Area Network)?

5. Що таке MAN (Metropolitan Area Network)?

6. Що таке WAN (Wide Area Network)?

7. Що таке корпоративна мережа підприємства (Enterprise Network)?

8. Які з протоколів використовує корпоративна мережа підприємства?

9. Що таке MAP (Manufacturing Automation Protocol)?

10. Що таке TOP (Technical and Office Protocol)?

11. Що таке системи MAP/TOP?

12. Яке основне призначення ЛОМ?

13. На яких принципах система розподілених ресурсів повинна ґрунтуватися?

14. Хто висунув концепцію архітектури відкритих систем і на чому вона повина ґрунтуватися?

15. Призначення протоколів?

16. На яки дві частини поділяється кожен рівень ЛОМ?

17. Що визначає специфікація послуг і специфікація протоколів?

18. Чи може кожний конкретний рівень мати більше одного протоколу?

19. Перерахуйте рівні управління і протоколи ЛОМ.

20. Перерахуйте функції фізичного рівня.

21. Перерахуйте функції канального рівня.

22. Перерахуйте функції мережного рівня.

23. Що таке маршрутизатори?

24. Що таке датаграма?

25. Що таке віртуальні з'єднання?

26. Перерахуйте функції транспортного рівня.

27. Перерахуйте функції сеансового рівня.

28. Перерахуйте функції представницького рівня.

29. Перерахуйте функції прикладного рівня.

30. Чи різняться одиниці виміру інформації на різних рівнях?

31. На якому принципі базується організація ЛОМ?

32. Що визначає протокол УФК (управління фізичним каналом)?
33. Що установлює протокол УДК (управління доступом до каналу)?
34. Що забезпечує протокол УІК (управління інформаційним каналом)?
35. Що забезпечує протокол УІ (управління транспортним інтерфейсом)?
36. У залежності від способу організації обробки даних і взаємодії користувачів які виділяють два типи інформаційних систем?
37. Що виконує центральний процесор в ієрархічних мережах?
38. Переваги та недоліки ієрархічних систем?
39. Що таке клієнт і сервер у системах клієнт-сервер?
40. Які прийнято виділяти за організацією взамоєдії типи систем, що використовують метод клієнт-сервер?
41. Що таке рівноправна мережа? Переваги та недоліки.
42. Що таке мережа з виділеним сервером? Переваги та недоліки.

# 3. Локальні мережі. Мережеві технології.

## 3.1. Методи доступу до передавального середовища в локальних мережах

Організацію доступу до передавального середовища здійснює нижній підрівень канального рівня- MAC- підрівень (Metode Access Control).

Спосіб організації доступу станцій мережі до передавального середовища називається *методом доступу.*

Різними розробниками мереж розроблена значна кількість різноманітних методів доступу, які можна розділити на групи:

- за характером використання фізичного середовища- метод доступу до моноканалу або мережа з ретрансляцією (методу моноканалу відповідає одночасне розсилання повідомлення до всіх станцій мережі),
- за характером керування- централізоване або децентралізоване керування,
- за характером доступу – конкурентний або з передаванням повноважень.

Зупинимося на найбільш використовуваних методах доступу, покладених в основу технологій найвідоміших локальних мереж. Розрізняють такі методи доступу:

- *Циклічний* (тактові системи). Він реалізується в так званих тактових ситеиах і полягає в тому, що весь час передавання розподіляється на однакові часові проміжки (слоти) для всіх робочих станцій. Кожна станція може заповнити повідомленням свій слот.

Недоліки методу:

- неефективне використання каналу- багато порожніх слотів.
- Зі збільшенням кількості станцій ефективність мережі зменшується. Метод можна використовувати в мережі до 100 станцій.

- *Метод опитування*. Використовується централізоване керування. Один з пристроїв- головний (контролер мрежі). Він керує передаванням. Найпростіший варіант реалізується на базі циклічного опитування. Контролер по черзі надсилає кадр до станцій. Ті відповідають, надсилаючи в мережу інформацію для передавання, або спеціальний кадр, коли інформації немає. Потім опиту.ться наступна станція і т.д. Використовується метод в шинних та ефірних мережах невелики розмірів.

Недоліки методу:

- наявність великого потоку керування навіть, коли у абонента немає інформації,
- надійність мережі визначається надійністю контролера,
- мережа обмежена щодо кількості абонентів. Приклад застосування- мережа MIC 1553B.

- *Метод кокурентного доступу*. Цей метод набув поширення спочатку в шинних мережах. Тут реалізовано принцип “слухай перш ніж говорити”, а саме –прослуховується канал зв’язку на наявність сигналу-носія, і лише при відсутності в каналі сигналу станція починаю передавати. При наявності сигналу чекає звільнення каналу.Цей метод отримав назву метода доступу з контролем сигналу –носія (МДКН), або Carrier Sencse Multiple Access (CSMA). В цьму методі можливі колізії між кадрами різних станцій завдяки кінцевій швидкості поширення електричного сигналу. В проміжку часу між початком передавння однією станцією та моментом спостреження цього

сигналу іншою станцією, ця остання може почати передавання свого кадру. Станції, які не передали свої кадри внаслідок колізії, знову пробують передати інформацію. Ефективність передавання при цьому не перевищує 53%. Якщо приймаються заходи проти колізій повторних передавань, то ефективність досягає 93%. Тут використовується процедура CD (Collision Detection), суть якої полягає в часовому розмежуванні повторних передавань. Удосконалений таким способом метод отримав назву CSMA/ CD.

- *Маркерний метод доступу*. В цьому методі в мережу вводять маркер-спеціальний кадр, який передається від станції до станції почерово по логічному кільцю. Станція, яка в конкретний момент має маркер, одержує право на передавання. Вона передає спочатку інформаційний кадр, а потім маркерний кадр. Якщо інформації немає, то передається маркерний кадр. Метод призначено длдя кільцевої, шинної, зіркової та деревоподібної топологій (Arcnet, Token Ring, Rignet…). У кільцевій мережі інформацію приймають, аналізують і передають до сусідньої станції, тобто здійснюється ретрансляція. Це дозволяє підсилювати сигнал. Станція –одержувач, отримавши інформаційний кадр, приєднує до кадру свій інформаційний кадр про отримання інформації, який передається на всі станції. Коли кадр з інформацією доходить до станції- відправника, ця станція звільняє кадр від інформаційного кадра і маркер знову передається по мережі.
- *Метод доступу з запитом пріорітету*. Цей метод використовують в мережах з топологією розгалуженого дерева. Центром кожної зірки є комутатор. Він опитує свої вихідні порти про наявність інформації для передавання. Завдяки відсутності колізій ефективність досягає 95%.

## 3.2. Мережеві технології

Під мережевими технологіями будемо розуміти технології обміну даними, реалізованими в найбільш використовуваних локальних мережах.

### 3.2.1. Технологія Ethernet

Перший лабораторний варіант мережі *Ethernet* було реалізовано в 1975 р. Це мережа з шинною топологією з CSMA/ CD. Стандартизована в стандартах IEEE- 802.3 та ECMA-82. Завдяки простоті, дешевості, здатності до масштабування *Ethernet* є лідером серед інших типів локальних мереж. Ця технологія продемонструвала значний потенціал розвитку і стла основою для технологій комутованого *Ethernet, Fast Ethernet Gigabit Ethernet*. Розроблено кілька типів кабельного з'єднання. Їх загальне позначення-

NNNN BASE-XX,

Де перші чотири цифри характеризують швидкість передавання в Мбіт/с, а символи XX – максимальну довжину сегмента в сотнях метрів або середовище передавання. Мережа *Ethernet* складається з одного або кількох сегментів, з'єднаних єа допомогою повторювачів або концентраторів. Приклади існуючих мереж *Ethernet*:

- 10 BASE –5- товстий *Ethernet*

  Топологія шинна. Максимальна довжина сегмента- 500 м., швидкість передавання 10Мбіт/с, сполучення- через товстий коаксіальний кабель,

- 10 BASE-2- тонкий *Ethernet*

Шинна багато сегментна топологія з довжиною сегмента 185 м. Сполучення- через тонкий коаксіальний кабель,

- 10 BASE-T(twisted)

Топологія- розподілена зірка (Рис.3.1). Сполучення –через скручену пару. Відстань до концентратора- до 165 м.


Рис.3.1.

Завдяки наявності концентраторів обмеження на розміри мережі незначні,

- 10 BASE-F(fibre)

Мережа на оптоволоконному кабелі. Довжина – до 2 км..Забезпечує тільки двопунктове з'єднання. Тому використовується тільки для магістральних ліній як доповнення до мережі на скрученій парі.

Типова мережа *Ethernet* передбачає приєднання 1024 робочих станцій в діаметрі до 1000 м. (з повторювачами). Кадр має наступну структуру :

| Преамбула | Адреса одержувача | Адреса від-правника | Тип | Дані | CRC |
|---|---|---|---|---|---|

Рис.3.2

Мінімальна довжина кадру 64 байти, максимальна- 1518 байт. З огляду на вплив колізій мінімальна довжина кадру вибрана з міркувань, щоб вона не була меншою подвоєної тривалості поширення сигналу між двома найвіддаленішими станціями. Таким чином, мінімальна довжина кадру обмежує діаметр мережі. Для надто великих довжин (понад 1500 байт) значно зростає вплив колізій, а точніше- ускладнюється боротьба з колізіями повторних передавань.

Хоча *Ethernet* має невелику перепускну здатність, немає фіксованої довжини кадрів, та завдяки технологіям комутації локальних мереж в віртуальних локальних мереж вказані недоліки мало суттєві. Крім того набуває широкого поширення швидкий *Ethernet* -

*Fast Ethernet*

Ця технологія була розроблена в 1992 р. Топологія – розподілена зірка. Кабелі – скручена пара, з'єднувачі – концентратори. Можуть бути і оптоволоконні кабелі. Відстань від концентратора до станції – 100 м. ,між станціями – до 210 м. Між двома станціями – не більше двох повторювачів. За допомогою стекових повторювачів ( з'єднання кількох повторювачів ), маршрутизаторів можно приєднати необмежену кількість сегментів *Fast Ethernet*. Розрізняють кілька типів мережі:

-100 Base –T – 100 Мбіт/с, скручена пара,
-100 Base –TX – дві пари скручених провідників категорії 5,
-100 Base –T4 – чотири пари скручених провідників,
-100 Base –F – на волоконно-оптичному кабелі.

Концентратори *Fast Ethernet* – це концентратори-повторювачі. Один тип – декодує сигнал і являється ретрансляційним повторювачем. В ньому можуть бути порти з різними форматиами *Fast Ethernet* .Другий тип – ретранслює аналоговий сигнал на всі порти. Два концентратори можно з'єднувати через спеціальні порти. Відстань – до 5м. В одному сегменті знаходяться концентратори одного типу. Обмеження на довжину сегмента та кількість концентраторів зв'язано з величинами затримок в розповсюдженні сигналу: затримки в концентраторі та затримки , зв'язаної з обробкою колізій.
Приклад структури мережі *Fast Ethernet* (Рис.3.3)

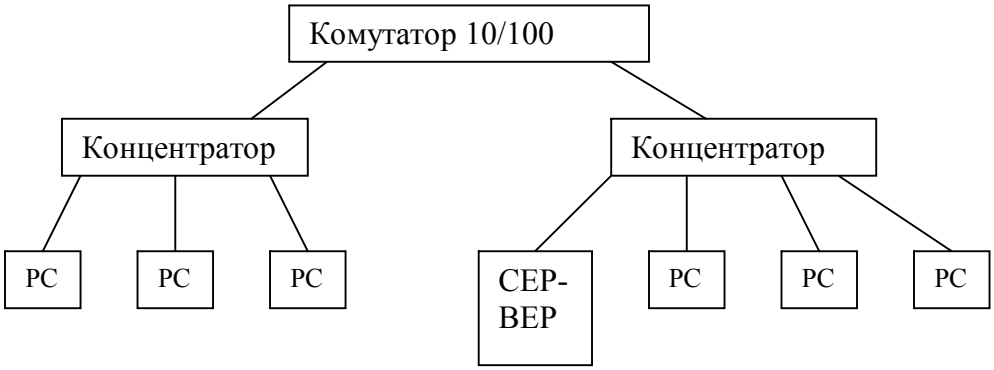



Рис.3.3.

Тут цифрами вказані швидкості передавання в Мбіт/с.

Мережа *Gigabit Ethernet* .

Мережа розроблена в 1998 р і стандартизована за стандартом IEEE-802.3z. Вона забезпечує швидкість передавання в 1000 Мбіт/с. Діють такі варіанти мережі:

- 1000 Base –SX –багатомодовий оптоволоконний кабель довжиною 275 м. (550 м. з повторювачем),
- 1000 Base –LX – багатомодовий та одномодовий оптоволоконний кабель. При одномодовому дальність досягає 5000 м.,
- 1000 Base –CX – коаксіальний кабель довжиною до 25 м.,
- 1000 Base –T - скручена пара довжиною до 100 м. Максимальний розмір домену – до 200 м.
- В мережі *Gigabit Ethernet* змінені параметри кадрів: мінімальна довжина кадру змінилась з 64 до 512 байт, максимальна – з 1500 до 9000 байт. Інтервал між кадрами змінився з 364 на 512 байт. Частота синхронізації-125 МГц. Сигнал має пять логічних рівнів (фазова модуляція з фазами $0^0$,$90^0$,$180^0$,$270^0$, $360^0$). Швидкість передавання – 125МГц*4=500 Мбод. Один імпульс передає 2 біти. Швидкість передавання даних – 500Мбод*2біт=1Гбіт/с.

Варіант структури мережі *Gigabit Ethernet* представлено на рис.3.4.

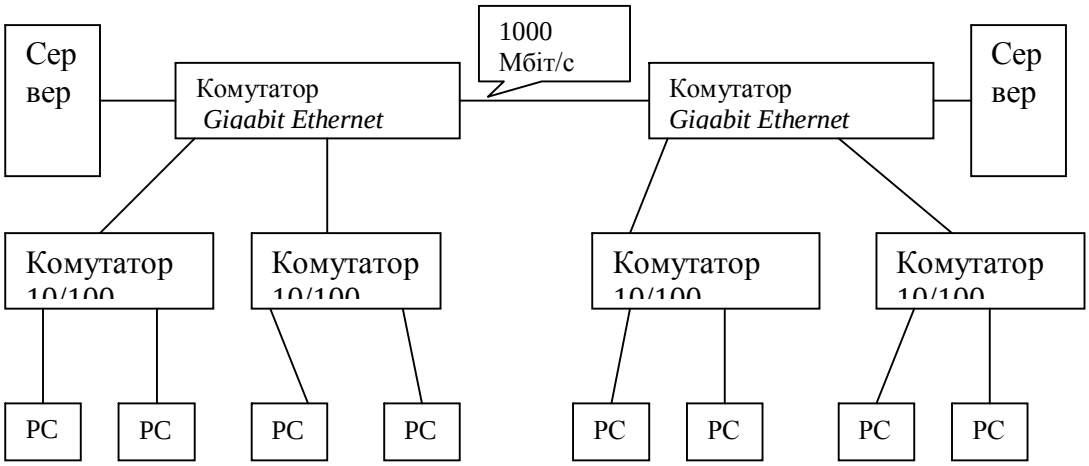



Рис.3.4.

### 3.2.2.Технологія *Token Ring*

Це мережа кільцевої топології з ретрансляцією та маркерним доступом. Розроблена в 60-х роках. Стандартизована у 1985 р. (IEEE-802.5) для швидкості 4 Мбіт/с. На сьогодні існує стандарт на 16 Мбіт/с. В 1998 р. розроблено стандарт IEEE-802.5t на швидкість 100 Мбіт/с Розробник мережі фірма IBM. Мережа посідає друге місце за *Ethernet* .При великих навантаженнях *Token Ring* (*TR* ) ефективніша ніж *Ethernet* ( в ситуації, коли *Ethernet* забезпечує лише 30-40% перепускної здатності, *TR* забезпечує 90% ефективності).

Варіант структури мережі *TR* представлено на Рис.3.5

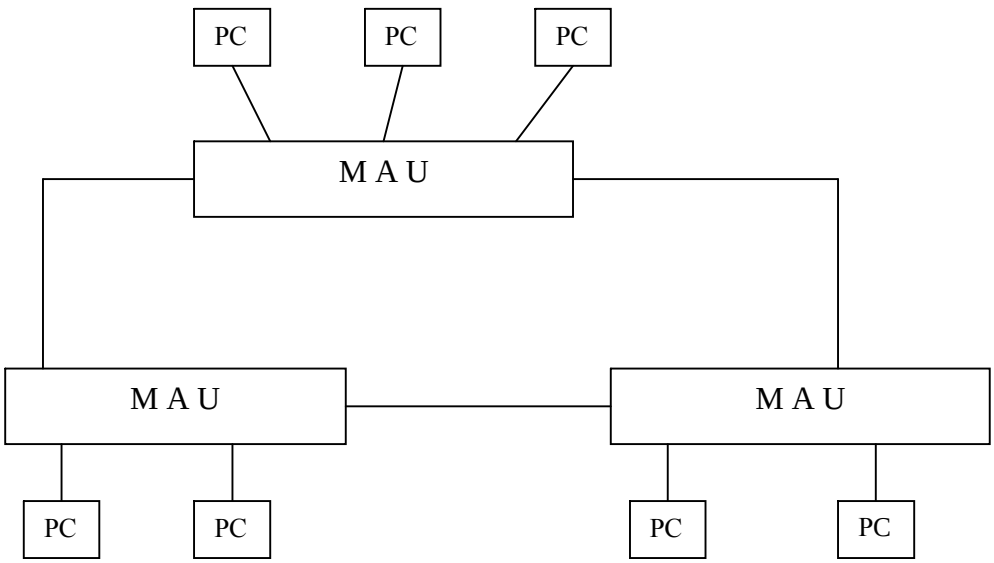



Рис.3.5

Пристрій MAU (Multistation Access Unit) –пристрій багатостанційного доступу. Кілька MAU з'єднуються в кільце. До MAU фізично під'єднується кілька робочих станцій, які можуть під'єднуватись до логічного кільця мережі або від'єднуватись. Від'єднування якоїсь робочої станції від мережі не впливає на роботу мережі. З'єднання здійснюється через спеціальні адаптери, в яких вбудовані мікросхеми виконують програмні функції керування передаванням даних у мережі. Комплект відповідних програм називають агентом. Агент взаємодіє на сеансовму рівні.

В *TR* використовується три типи адресації:

- індивідуальна (для кожної PC),
- групова (для передавання повідомлення для групи, яке потім циркулярно передається всім членам групи),
- функціональна – для деяких станцій, які крім загальних функцій виконують певні призначені функції (сервер звітів, монітор помилок кільця, сервер парамнетрів кільця і таке інше).

В мережі визначено три типи кадрів:

- кадр маркера,
- кадр даних,
- кадр послідовності аварійного завершення.

У мережі в будь-який час одна з робочих станцій являється головним менеджером кільця-так званий активний монітор (AM), всі інші станції являються пасивними моніторами- SM (Standby Monitor). Крім загального процесу передавання даних SM стежать за наявністю у мережі активного монітора. Якщо SM через деякий час не відшукують кадру AMP (Activ Monitor Present), вони запускають процедуру оголошення маркера, в результаті чого один з SM стає активним монітором. AM підтримує тактовий генератор мережі, періодично передає кадри AMP, що дає змогу користувачам від'єднувати станції з кільця, перевіряє цілісність кільця та правильність передавання кадрів.

Мережа *TR* дещо складна в своєму функціонуванні. Тому вона значно поступається мережі *Ethernet.*

Дещо покращений варіант мережі *TR* забезпечує швидкість передавання в 16 Мбіт/с. Тут довжина кадру з 4500 байт змінена на18000 байт. Використано випереджувальне звільнення маркера, тобто станція- відправник не чекає повернення до неї інформаційного кадру, а передає маркер сусіду зразу після інформаційного кадру.

Останнім часом розроблена новіша версія *Token Ring – Higth –Speed Token Ring*, що забезпечує швидкість передавання до 155 Мбіт/с

### 3.2.3. Інші технології

♦ Локальна мережа *Arcnet* (Attached Resource Computer).

Мережа *Arcnet* – маркерна локальна мережа зіркової або шинної топології, розроблена на початку 70-х років фірмою Dufapoint. Передача маркера відбувається від однієї станції до іншої в порядку зменшення їх логічних адрес. Станція з мінімальною адресою передає кадр маркера станції з найбільшою адресою. Керує мережею станція, яка володіє в даний момент маркером. Вона виконує:

- генерацію (реконфігурацію) логічного кільця,
- контроль за передачею маркера,
- зміну параметрів системи управління,
- прийом та обробку запитів на підключення пасивних станцій (станцій, що не під'єднані до логічного кільця).

Швидкість передавання досягає 2,5 Мбіт/с. Довжина кадру-508 байт. 8-и розрядні адреси не відповідають стандартам IEEE.

Найпоширенішою є зіркова топологія. Робочі станції приєднані до концентраторів, які можуть бути активними і пасисвними. Активні підсилюють сигнал, мають зовнішнє живлення. До них можна приєднати до 8-и станцій на відстані до 600 м. До пасивного- 4 станції на відстані 30 м.

Топологія шинної мережі подібна мережі *Ethernet* –10 Base-2. Адаптери Arcnet дешевші від *Ethernet.* До однієї шини можна приєднати до 8-и станцій.

Мережа *Arcnet* – найдешевша і найгнучкіша за технологією. Але *Ethernet* витісняє її завдяки своїм новим швидісним версіям .

- Локальна мережа *FDDI* (Fiber Distributed Data Interface).

  Ця мережа належить до перспективних вирішень. За топологією *FDDI* являєтьсяподвійним оптоволоконним кільцем Швидкість передавання – до 100 Мбіт/с. Кожне кільце має довжину до 200 км.. Вузли – на відстані до 2,5 км. Максимальна кількість станцій- 1000. Мережу *FDDI* розглядають як перехідну ЛМ до міської мережі (MAN). Метод доступу- маркерний з випереджувальним звільненням маркера. Топологія мережі кільцева або деревоподібно-кільцева. Під'єднування РС до вузла (концентратора) можливо і за допомогою скрученої пари.
- Перспективною технологією є технологія ATM (Acinchronous Transfer Mode), яка в перспективе може стерти відмінності між локальними та глобальними мережами. Про неї детальніше буде далі.

### 3.3 Питання для самоконтролю

1.Що таке метод доступу до мережі?

2. Які існують методи доступу?
3. Що таке маркерний метод доступу?
4. Що таке конкурентний метод доступу?
5. Що таке колізії в мережі?
6. Які приймаються заходи проти колізій?
7. Що таке мережева технологія?
8. Що таке мережа *Ethernet*?
9. Що таке мережа *Fast Ethernet*?
10. Що таке мережа *Gigabit Ethernet*?
11. Що таке мережа *Token Ring*?
12. Яке маркування використовується для опису різних варіантів мережі *Ethernet*?
13. Яке маркування використовується для опису різних версій *Fast Ethernet?*
14. Що таке мережа *Arcnet?*

15. Що таке мережа *FDDI?*.

# 4. ІНТЕРНЕТ

## 4.1. Основні поняття

*Інтернет* — це об'єднання мереж. У останні роки у цього поняття з'явився ширший зміст: *Всесвітня комп'ютерна мережа.* Інтернет можна розглядати не тільки як об'єднання мільйонів комп'ютерів лініями зв'язку, а як інформаційний простір для розв'язання будь яких інформаційних і обчислювальних задач.

Дані, що відправляють користувачі зі свого комп'ютера розділяються на пакети, які можуть в одному сеансі зв'язку пройти до місця призначення різними шляхами. Якими маршрутами пакети одного повідомлення не мандрують, але вони приходять до місця призначення і об'єднуються в єдиний документ. При цьому дані, що відправлені раніш, можуть приходити пізніше і навпаки. Ця ситуація не заважає зібрати правильно документ тому, що кожний пакет має своє маркерування.

У різних системах комп'ютерних мереж пакети даних визначаються по-різному, але загальними для них є такі елементи:

- унікальна адреса відправника;
- унікальна адреса одержувача;
- ознака, що характеризує зміст пакета; дані або повідомлення;
- контрольна сума (CRC) для виявлення помилок при передачі даних.

Базова схема пакета повідомлень має такий вигля

| Адреса відправника | Адреса одержувача | Тип пакета | Дані/Повідомлення | CRC |
|---|---|---|---|---|

Таким чином, Інтернет можна представити як "простір", в якому здійснюється неперервна циркуляція даних. Ці дані переміщуються між комп'ютерами, що складають вузли мережі, і зберігаються на їх жорстких дисках потрібний час.

## 4.2. Історія розвитку Інтернету

В 50-х роках XX століття почалися експерименти по прийому та передачі даних за допомогою комп'ютерів, які мали лабораторний характер.

В 1958 році США прийняли рішення про створення першої глобальної мережі національного масштабу. Розробка мережі здійснювалася Пентагоном для раннього оповіщення про запуск ракет (NORAD — North American Aerospace Defense Command). Станції системи NORAD протягнулися через північ Канади від Аляски до Гренландії, а підземний пункт управління був розташований в горі Шайенн поблизу міста Колорадо -Спрингс.

В 1964 році був введений центр управління системою NORAD, до якого стали підключатися авіаційні, метеорологічні і інші військові та цивільні служби. Керуванням роботою мережі зайнялася спеціальна організація — Управління перспективними розробками міністерства оборони США (DARPA — Defense Advanced Project Agency). Недоліком цієї мережі була її недостатня стійкість при виході з ладу будь якого вузла або командного пункту.

В 1969 році була розроблена національна комп'ютерна мережа APRANET між університетами і науковими центрами США.

В 1983 році здійснилися революційні зміни в програмному забезпечені комп'ютерного зв'язку. Проблема стійкості глобальної мережі була розв'язана застосуванням протоколу TCP/IP, якій є основою всесвітньої мережі і в даний час.

В другій половині 80-х років здійснилося ділення всесвітньої мережі на домени (DNS — Domain Name System). Коли з'явилася система доменних імен, то тоді з'явилося поняття Інтернету як децентралізованої ієрархічної структури, що сама розвивається.

## 4.3. Протоколи TCP/IP та служби Інтернет

Мережа Інтернет (Internet) ґрунтується на моделі взаємодії відкритих систем (OSI) з застосуванням протоколів TCP/IP. В технічному розумінні

протокол TCP/IP складається з двох протоколів, які знаходяться на різних рівнях *стека протоколів* Протокол TCP (Transmission Control Protocol) — транспортного рівня, що визначає, як відбувається передача інформації. Протокол IP (Internet Protocol) — адресний протокол мережного рівня, що визначає, куди передається інформація.

Практично всі послуги мережі побудовані на принципі "клієнт-сервер".

*Сервер* (у мережі Internet**) —** це комп'ютер або програма, які здатні надавати клієнтам (у міру надходження від них запиту) деякі мережні послуги.

*Клієнт* — прикладна програма, завантажена в комп'ютер користувача, яка забезпечує передачу запитів до сервера й одержання відповіді від нього.

По мірі розвитку мережі з'являються нові протоколи (сервіси), змінюючи її вигляд і розширюючи коло користувачів. Щоб скористатись якоюсь із служб мережі Internet, необхідно встановити на ПК клієнтську програму . здатну працювати за протоколом цієї служби. Деякі клієнтські програми входять до складу ОС Windows 98, NT, програм – броузерів ( наприклад, Microsoft Internet Explorer та Netscape Communicator).

*Сервіс* FTP (File Transfer Protocol) дає можливість обмінюватися двійковими і текстовими файлами між ПК мережі. Для вузлів FTP характерною є наявність процедури входу (login). Як "гостьові" ім'я і пароль часто використовують імена anonymous, ftp та адреса електронної пошти. Популярна клієнтська програма FTP забезпечує зручний Windows–подібний інтерфейс для FTP-сервісу.

*Електронна пошта* (E-mail) — забезпечує обмін поштовими повідомленнями між ПК мережі Internet. Серед клієнтських поштових програм можна виділити TheBat!, Eudora Pro.

*Сервіс* Mail Lists (списки розсилки) створений на підставі протоколу електронної пошти забезпечує можливість регулярно одержувати електронною поштою повідомлення про науково-технічні й економічні огляди, презентації

нових програмних та апаратних засобів обчислювальної техніки і т. д.

*Сервіс Usernet* (групи новини або телеконференції) — забезпечує обмін інформацією між усіма, хто ним користується. Новини поділяються за темами на групи, що якоюсь мірою їх упорядковує. На певні групи можна оформити підписку. Перегляд інформації за темами називається *моніторингом інформації.* Для реалізації цього сервісу існують клієнтські програми, наприклад, Microsoft Internet News.

*Сервіс* WWW (World Wide Web — всесвітня павутина). Сервіс WWW — це єдиний інформаційний простір, який складається з сотень мільйонів взаємозв'язаних гіпертекстових електронних документів, що зберігаються у Web-серверах. Окремі документи називаються Web-сторінками. Групи тематично об'єднаних Web-сторінок утворюють Web-вузол (жаргонний термін — Web-сайт, або просто сайт).

Web-сторінка — це текстовий файл, що містить опис або зображення мультимедійного документа на мові гіпертекстової розмітки — HTML (Hyper-Text Markup Language). Сторінка також може містити графічні, звукові та відео об'єкти. З будь яким фрагментом тексту (наприклад, із малюнком) можна пов'язати інший Web-документ, тобто встановити гіперпосилання. У цьому разі під час клацання лівою клавішею миші на тексті або малюнку, що є гіперпосиланням, відправляється запит на доставку нового документа. Цей документ, у свою чергу, також може мати гіперпосилання на інші документи.

Для передачі інформації у WWW використовується протокол http (hypertext transfer protocol — протокол передачі гіпертексту).

Перегляд Web-сторінок і переміщення через посилання користувачі здійснюють за допомогою програм броузерів ("to browse"– переглядати). Найбільш популярними Web-броузерами в Україні є Microsoft Internet Explorer, Netscape Communicator, OPERA.

*Сервіс* IRC (Internet Relay Chat) забезпечує проведення телеконференцій в реальному часі. Іноді службу IRC називають чат-конференціями, або просто

чатом. Існують популярні клієнтські програми, що підтримують сервіс IRC, наприклад, mIRC, mIRC32 для Windows. Вони застосовуються для проведення групових консультацій і нарад.

*Служба* ICQ призначена для пошуку мережної IP-адреси людини, ПК якої приєднано в даний момент до мережі Internet. При кожному приєднанні до мережі Internet програма ICQ, встановлена на ПК користувача, визначає поточну IP-адресу і повідомляє його центральній службі, яка в свою чергу, оповіщає партнерів користувача. Далі партнери можуть встановити з користувачем прямий зв'язок. Після встановлення контакту зв'язок відбувається в режимі, аналогічному сервісу IRC.

*Сервіс Telnet* (віддалений доступ) дає можливість абоненту працювати на будь-якій ЕОМ, як на своїй власній. Часто протоколи Telnet застосовують для дистанційного керування технічними об'єктами (телескопами, відеокамерами і таке інше). Прикладом програми, що реалізує доступ до Telnet-сервісу, може бути програма Net Term.

Найближчим часом плануються: впровадження таких серверів, як Real Audio та Real Video; приєднання до мережі через кабельні мережі телебачення; Web-телефонія; застосування інтерактивних Web-сторінок, які стануть виконувати функції не тільки бази даних, а й функції програм, і багато що інше.

## 4.4. Підключення до Інтернету

Для роботи в Інтернеті необхідно:

- фізично підключити комп'ютер до одного із вузлів Всесвітньої мережі;
- придбати IP-адресу на постійній або тимчасовій основі;
- встановити і налаштувати програмне забезпечення — програми-клієнти тих служб Інтернету, послугами яких передбачається користовуватись.

Організації, що дають можливість підключитися до свого вузла і виділяють IP-адресу, називаються *постачальниками послуг Інтернету* (використовується ще термін *сервіс-провайдер*). Цю послугу дають на договірній основі.

Фізичне підключення може бути *виділеним* або комутованимм. Для виділеного з'єднання прокладають нову або орендують готову фізичну лінію зв'язку (кабельну, радіоканал, супутниковий канал тощо). Для комутованого з'єднання застосовують телефону лінію.

Серед характеристик комунікаційної мережі найважливішими є:

- швидкість передачі даних по каналу зв'язку;
- пропускна здатність каналу зв'язку;
- вірогідність передачі інформації;
- надійність каналу зв'язку і передавальної апаратури.

В даний час пропускна здатність супутникових ліній зв'язку складає сотні мегабіт в секунду (Мбіт/с), а телефонних ліній — 60-120 кілобіт в секунду (Кбіт/с).

Підключення ПК до Інтернету здійснюється через телефонні лінії, за допомогою *модему*, який перетворює цифрові сигнали комп'ютера на аналогові сигнали телефонної лінії, та навпаки.

Модем потребує апаратної і програмної установки. В операційній системі Windows 98 її можна виконати стандартними засобами **Пуск/ Настройка/ Панель управления/Модемы.** Перевірку підключення модему можна здійснити командою **Пуск/Настройка/Модемы/Диагностика/ Дополнительно.**

Для підключення до комп'ютера постачальника послуг Інтернет потрібно настроїти програму **Удаленный доступ к сети (Мой компьютер/Удаленный доступ к сети/Новое соединение**). При цьому потрібні дані, які повинен повідомити постачальник послуг:

- номер телефону, по якому здійснюється з'єднання;
- ім'я користувача (login);
- пароль (password);
- IP-адресу сервера DNS;
- номер телефону служби підтримки (на всякий випадок).

Вводити власну IP-адресу не має сенсу тому, що сервер постачальника послуг виділить його автоматично на час проведення сеансу роботи.

IP-адреса користувача складається з його ідентифікатора, символу @, а потім кількох частин домену, відокремлених крапкою. Ці частини мають ієрархічну структуру, починаючи з самого нижчого рівня. Наприклад, IP-адреса Marach@kytep.kiev.ua розшифровується таким чином:

- Marach — назва комп'ютера користувача;
- кутер — назва організації;
- kiev — назва міста;
- ua — скорочена назва країни.

## 4.5. Питання для самоконтролю

1. Що таке Інтернет?
2. На що розділяються дані, що відправляють користувачі зі свого комп’ютера?
3. Які загальні елементи є в пакетах мережі?
4. Який вигляд має базова схема пакета повідомлень?
5. Історія розвитку Інтернету.
6. Яка проблема глобальної мережі була розв’язана застосуванням протоколу TCP/IP?
7. Коли здійснилося ділення всесвітньої мережі на домени?
8. Коли з’явилося поняття Інтернету як децентралізованої ієрархічної структури, що сама розвивається?
9. На чому ґрунтується мережа Internet?
10. Що таке стек протоколів?
11. Якого рівня протокол TCP і що він визначає?
12. Якого рівня протокол IP і що він визначає?
13. Що таке принцип “клієнт-сервер”?

14. Яка проблема була розв’язана в глобальної мережі при застосуванні протоколу TCP/IP?

15. Що таке сервер?

16. Що таке клієнт?

17. Яку можливість дає сервіс FTP?

18. Що забезпечує електронна пошта (E-mail)?

19. Що забезпечує сервіс Mail Lists (списки розсилки)?

20. Що забезпечує сервіс Usenet (групи новини або телеконференції)?

21. Із чого складається сервіс WWW (World Wide Web — всесвітня павутина)?

22. Що містить Web-сторінка?

23. Який протокол використовується для передачі інформації у WWW?

24. Що забезпечує сервіс IRC (Internet Relay Chat)?

25. Для чого призначена служба ICQ?

26. Яку можливість абоненту дає сервіс Telnet (віддалений доступ)?

27. Що необхідно для роботи в Інтернет?

28. Яким може бути фізичне підключення?

29. Які характеристики комунікаційної мережі є найважливішими?

30. Який пристрій перетворює цифрові сигнали комп’ютера на аналогові та навпаки?

31. Які установки потребує модем?

32. З яких частин складається IP-адреса користувача?

# 5. ОДЕРЖАННЯ ІНФОРМАЦІЇ З ІНТЕРНЕТ

## 5.1. Основні поняття

Інтернет використовується як джерело різноманітної інформації з різних галузей знань. Термін World Wide Web (Web, WWW) середовище WWW — це сукупність Web-документів, між якими є гіпертекстові зв’язки.

Середовище WWW не має централізованої структури і розглядається як інформаційний простір. Документи WWW зберігаються на Web-серверах. На

Web-серверах розміщують не окремий документ, а групу взаємопов'язаних документів. Така група має назву Web-вузол (або Web-сайт). Розміщення матеріалів на Web-вузлі має назву Web-видання або Web-публікація.

*Web-канали* — це такі Web-вузли, які мають можливість самостійно здійснювати постачання інформації. Концепція каналів підтримується операційною системою Windows 98. На їх основі здійснюється динамічне оновлення Робочого столу Active Desktop.

*Web-сторінка* — це окремий документ World Wide Web. Звичайно це комбінований документ, котрий може мати текст, графічні ілюстрації, мультимедійні і інші об'єкти, що вставляються. Для створення Web-сторінки використовується мова розмітки HTML (HyperText Markup Language), яка за допомогою *тегів*, що вставляються в документ, описує логічну структуру документа, керує форматуванням тексту і розміщенням об'єктів, що вставляються. *Інтерактивні* Web-сторінки отримують інформацію від користувача через *форми* і генерують запитану Web-сторінку за допомогою спеціальних програм (*сценаріїв CGI*), *динамічного* HTML і інших засобів.

*Гіперпосилання* — це засіб переходу від одного документа до іншого, який з ним тематично пов'язаний, без вказівки явної адреси. Зв'язок між документами здійснюється за допомогою *гіпертекстових посилань* (або просто *гіперпосилань*). Механізм гіперпосилань дозволяє організувати тематичну подорож по World Wide Web без використання адрес конкретних сторінок.

*Адресація документів* використовує форму, яку називають *адресою URL*. Адрес *URL* має вказівку на прикладний протокол передачі, адресу комп'ютера і шлях пошуку документа на цьому комп'ютері. Адрес комп'ютера складається з декількох частин, які розділяються крапками. Наприклад, www.intel.ru. Частини, що розташовані праворуч, визначають мережну належність комп'ютера, а ліві елементи вказують на конкретний комп'ютер даної мережі. Перетворення адреси *URL* на цифрову форму *IP*-адреси виконує служба імен доменів (DNS, Domain Name Service). В якості роздільника в шляху пошуку документа Інтернет завжди застосовується символ косої риски (/).

*Засоби перегляду Web.* Мова HTML забезпечує не стільки форматування документу, скільки описання його логічної структури. Форматування і відображення документа на конкретному комп'ютері виконується спеціальною програмою — *броузером* (browser).

Основні функції броузера такі:

- встановлення зв'язку з Web-сервером, на якому зберігається документ, і завантаження усіх компонентів комбінованого документа;
- інтерпретація тегів мови HTML, форматування і відображення Web-сторінки у відповідності з можливостями комп'ютера, на якому працює броузер;
- надання засобів для відображення мультимедійних і інших об'єктів, які входять до складу Web-сторінки, а також механізму розширення, який дозволяє налагоджувати програму на роботу з новими типами об'єктів;
- забезпечення автоматизації пошуку Web-Сторінки і спрощення доступу до Web-сторінок, що розглядалися раніше;
- надання доступу до вбудованих або автономних засобів для роботи з іншими службами Інтернет.

## 5.2. Робота з програмою Internet Explorer 5.0

Схема використання Інтернет через Internet Explorer (рис. 5.1.).

З боку Інтернет роботу служби World Wide Web забезпечують програмні засоби Web-сервера. З боку користувача роботу забезпечують клієнтські програми Web-броузера. Найбільш відомими броузерами є Internet Explorer (компанія Microsoft), Netscape Navigator (компанія Netscape Communications) і Opera (компанія Opera Software). У броузера Microsoft Internet Explorer є деякі

переваги в тому, що, починаючи з ОС Windows 98, він постачається разом з системою і є її компонентом.

Програма Microsoft Internet Explorer пропонує єдиний метод доступу до локальних документів комп'ютера, ресурсів корпоративної мережі Intranet і до інформації, що доступна в Інтернеті. Вона забезпечує роботу з World Wide Web,

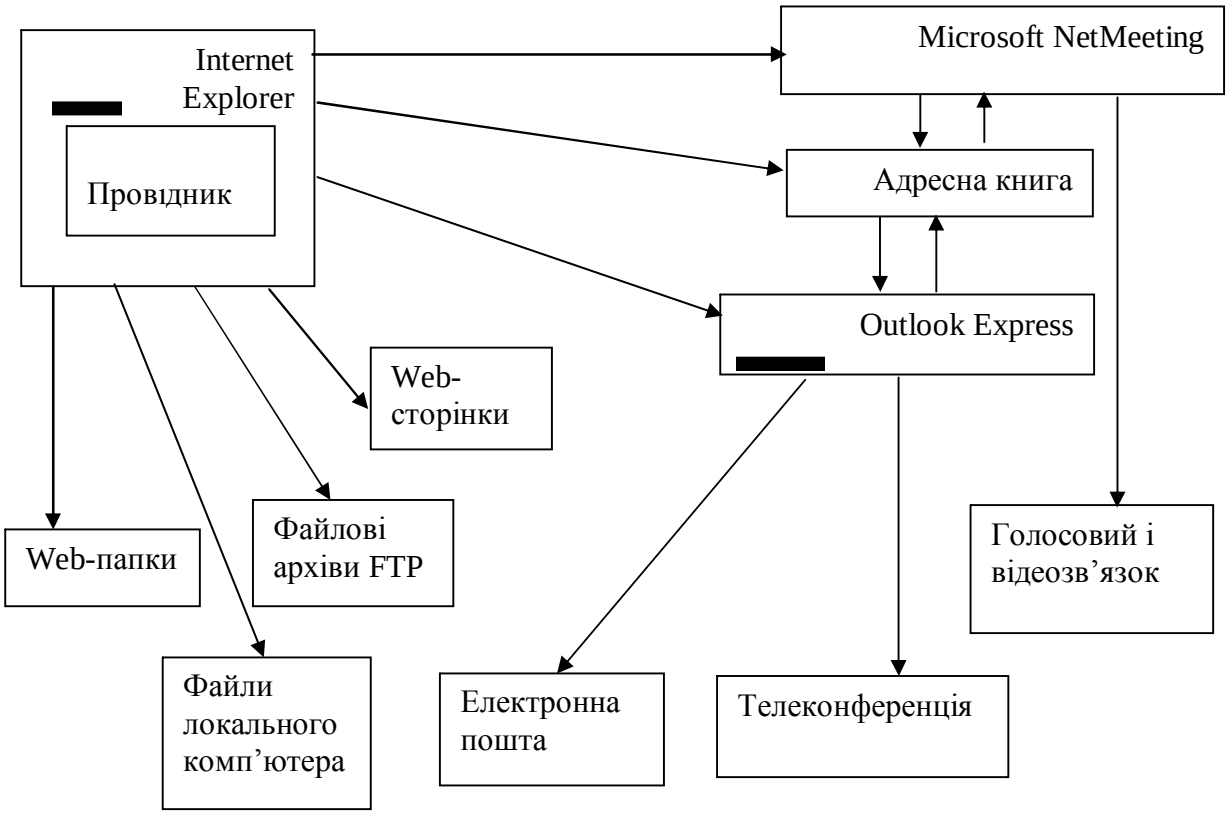



**Рис. 5.1.** Організація доступу до ресурсів Інтернет

рекомендує ідентичні засоби роботи з локальними папками комп'ютера і файловими архівами FTP,дає доступ до засобів зв'язку через Інтернет. Відповідні програми (Outlook express і Microsoft NetMeeting) автономні, але розглядаються як частина пакету Microsoft Internet Explorer 5.0.

Для запуску броузера Internet Explorer можна використовувати значок Internet Explorer на Робочому столі або на Панелі швидкого запуску, або виконуючи команду **Пуск/Программы/Internet Explorer.** Програма може запускатися автоматично при спробі відкрити документ Інтернету або локальний документ в форматі HTML. Для цього можна використати ярлики Web-сторінок, папку **Избранное (Пуск/Избранное** або меню **Избранное** в рядку меню вікна папки

або програми **Проводник**), панель інструментів Робочого столу **Адрес** або поле вводу в діалоговому вікні **Запуск программы (Пуск/Выполнить)**.

Коли з'єднання з Інтернетом відсутнє, то після запуску програми на екрані з'являється діалогове вікно для управління установкою з'єднання. При неможливості установити з'єднання зберігається можливість перегляду в *автономному режимі* Web-документів, що раніше були завантаженні. При присутності з'єднання після запуску програми на екрані з'являється *початкова* сторінка, яка вибирається при налагоджуванні програми.

## 5.3. Відкривання і перегляд Web-сторінок

Web-сторінка, що переглядається, відображається в робочій області вікна.

За замовчуванням завантажується увесь зміст Web-сторінки разом з графічними ілюстраціями й вбудованими мультимедійними об'єктами (рис.5.2.). Управління переглядом здійснюється за допомогою рядка меню, панелей інструментів, а також активних елементів, що знаходяться у відкритому документі (наприклад, гіперпосилань).

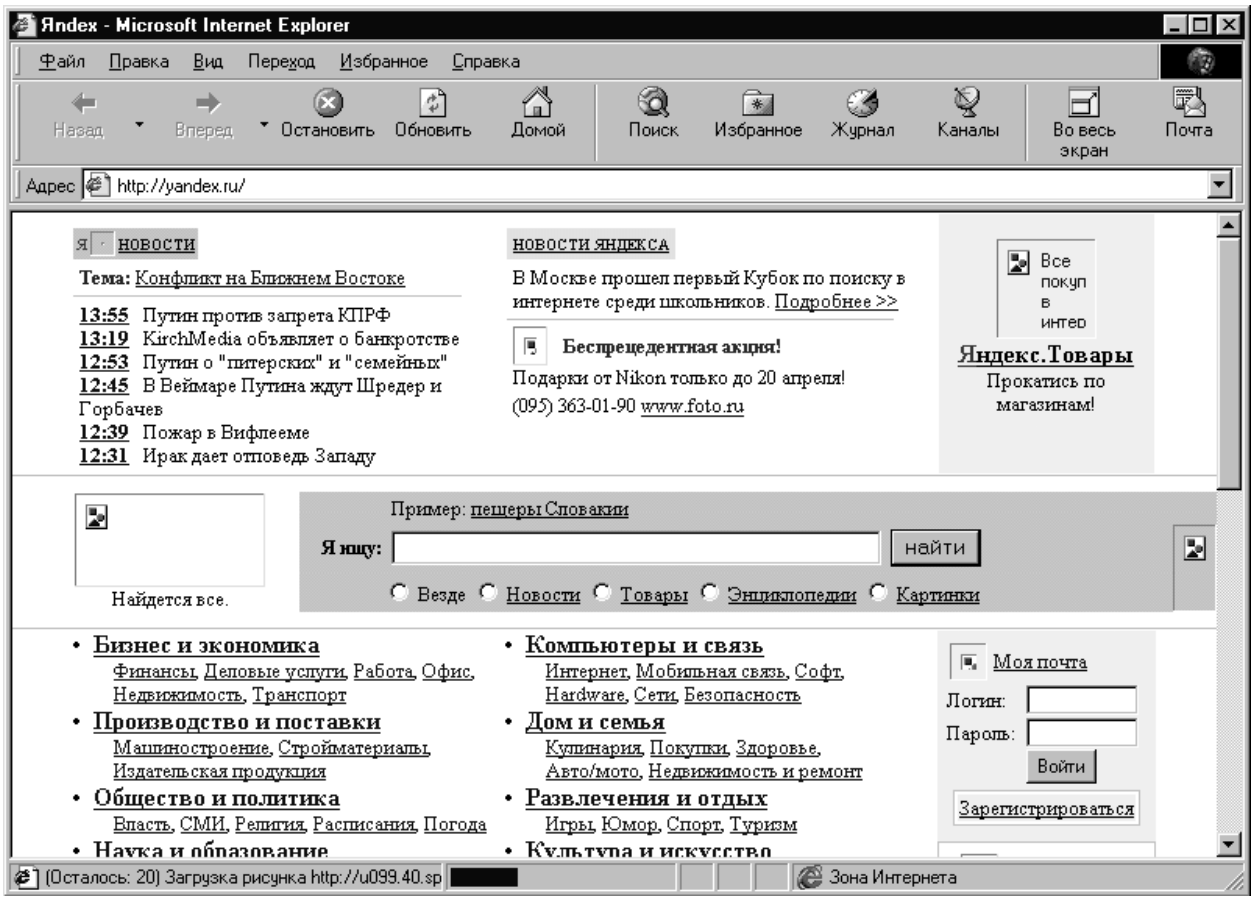



**Рис. 5.2.** Web-сторінка в період завантаження

Коли URL-адреса Web-сторінки відома, то можна ввести в поле панелі **Адрес** і клацнути на кнопці **Переход**. Сторінка з адресою відкривається замість поточної сторінки. Спосіб автозаповнення адресного рядка спрощує повторне введення адрес. Введена адреса автоматично порівнюється з адресами Web-сторінок, що розглядалися раніше. Всі доступні адреси відображаються в списку панелі **Адрес**, що розкривається. Коли потрібна адреса є в списку, то її можна вибрати клавішами-стрілками (↑, ↓), після чого клацнути на кнопці **Переход**. При відсутності потрібної адреси введення її здійснюється звичайним чином.

**Робота з гіперпосиланнями.** При мандруванні по Інтернету часто використовують гіперпосилання, які виділені кольором (наприклад, синім) і підкресленням. Підкреслення застосовується тільки для виділення гіперпосилань. При наведенні курсором миші на гіперпосилання він приймає

вигляд кисті руки, а саме гіперпосилання при відповідному налаштуванні броузера змінює колір. Адреса URL, на яку вказує посилання, відображається в рядку стану. При клацанні на гіперпосилання відповідна Web-сторінка завантажується замість поточної. Коли гіперпосилання вказує на довільний файл, його завантажування здійснюється за протоколом FTP.

На Web-сторінках можуть також зустрічатися графічні посилання, які об'єднують декілька посилань в рамках одного зображення. Для перегляду посилань зручно використовувати кнопку [Tab]. При натисненні на цю кнопку фокус вводу (пунктирна рамка) переміщується до наступного посилання. Зробивши перехід по посиланню, натискують клавішу [Enter]. Таким чином можна послідовно перебирати текстові і графічні посилання, а також окремі області зображень-карт.

Додаткові можливості використання гіперпосилань пропонує їх контекстне меню. Щоб відкрити нову сторінку, не закриваючи поточній, застосовують команду **Открыть в новом окне**. В цьому випадку відкривається нове вікно броузера. Адресу URL, що задана посиланням, можна розташувати в буферу обміну за допомогою команди **Копировать ярлык.** Адресу можна вставити в полі панелі **Адрес** або в будь-який документ для подальшого використання.

Інші операції, що відносяться до поточної сторінки і її елементів, також зручно виконувати через контекстне меню. Коли рисунок виконує функції графічного посилання, до нього можна застосовувати як команди, що відносяться до зображення, так і команди, що відносяться до посилання.

*Способи управління броузером.* Для перегляду документів World Wide Web зручно використовувати кнопки панелі інструментів **Обычные кнопки.**

Кнопка **Назад** використовується для повернення до сторінки, яка переглядалася деякий час назад.

Кнопка **Вперед** дозволяє відмінити дію, що виконана за допомогою кнопки **Назад.**

Кнопка **Остановить** використовується, коли процес завантаження сторінки затягнувся або необхідність в ній відпала.

Кнопка **Обновить** дозволяє заново завантажити сторінку, коли її завантаження було перервано або зміст документа змінився.

Кнопка **Домой** використовується для завантаження початкової сторінки, з якої броузер починає свою роботу.

Команди меню **Файл** дозволяють створити нове вікно, зберегти відкритий документ, надрукувати його, включити або виключити режим автономної роботи, а також завершити роботу з програмою.

Команди меню **Правка** дозволяють виконати копіювання фрагментів документа в буфер обміну і пошук тексту на Web-сторінці.

Команди меню **Вид** дозволяють включення і виключення зображення службових документів вікна (панелей інструментів, додаткових панелей, рядка стану), вибір шрифту і кодування символів.

Команди меню **Избранное** дозволяють ведення списку сторінок, які регулярно відвідуємо, і швидкий доступ до них.

Команди меню **Сервис** дозволяють перехід до використання програм для роботи з іншими службами Інтернет, а також настроювання броузера.

**Робота з декількома вікнами**. Для відкриття нового вікна програми Internet Explorer використовують команду Файл/Создать/Окно. Кожне вікно відображає свій Web-документ і може використовуватися самостійно. Списки кнопок Назад і Вперед оновлюються в кожному вікні самостійно.

Закрити вікна програми Internet Explorer можна в будь-якому порядку, а не тільки в тому, як вони відкривалися. При закритті останнього вікна на комп'ютері може більш не бути відкритих програм, що використовувалися в Інтернеті. В цьому випадку на екран видається повідомлення, яке дозволяє розірвати з'єднання, коли воно більше не потрібно.Налаштування властивостей броузера**.**

**Налаштування броузера,** необхідне для ефективної роботі в Інтернеті і залежить від *багатьох факторів*:

- властивостей відеосистеми комп’ютера;
- продуктивності діючого зв’язку з Інтернетом;
- змісту поточного Web-документа;
- особистих переваг індивідуального користувача.

Почати настроювання програми Internet Explorer можна як із самої програми (**Сервис/Свойства обозревателя**), так і через Windows — **Панель управления** (значок **Свойства обозревателя**). Вікно діалогу, що відкривається, має в цьому випадку лише різні назви: **Свойства обозревателя** або **Свойства: Интернет** (рис. 5.3.). Це вікно містить шість закладок для налаштування різних груп параметрів.

Загальні параметри роботи броузера задаються на закладці **Общие**. На цій закладці можна вказати, яку сторінку треба використовувати як основну, задати об’єм дискового простору для збереження тимчасових файлів, що прийняті із Інтернет, і вилучити тимчасові файли, а також сторінки, що підготовлені для читання в автономному режимі. Правила збереження тимчасових файлів задаються за допомогою кнопки **Настройка**, яка викликає вікно **Настройка** (рис. 5.4.). Кнопка **Обновить** на панелі інструментів **Обычные кнопки** дозволяє отримати останню версію документа незалежно від настроювань. На закладці **Общие** кольори встанвлюються за допомогою кнопки **Цвета**, а шрифти — за допомогою кнопки **Шрифты**. Ці налаштування підлеглі тому, що задаються в Web-документі.

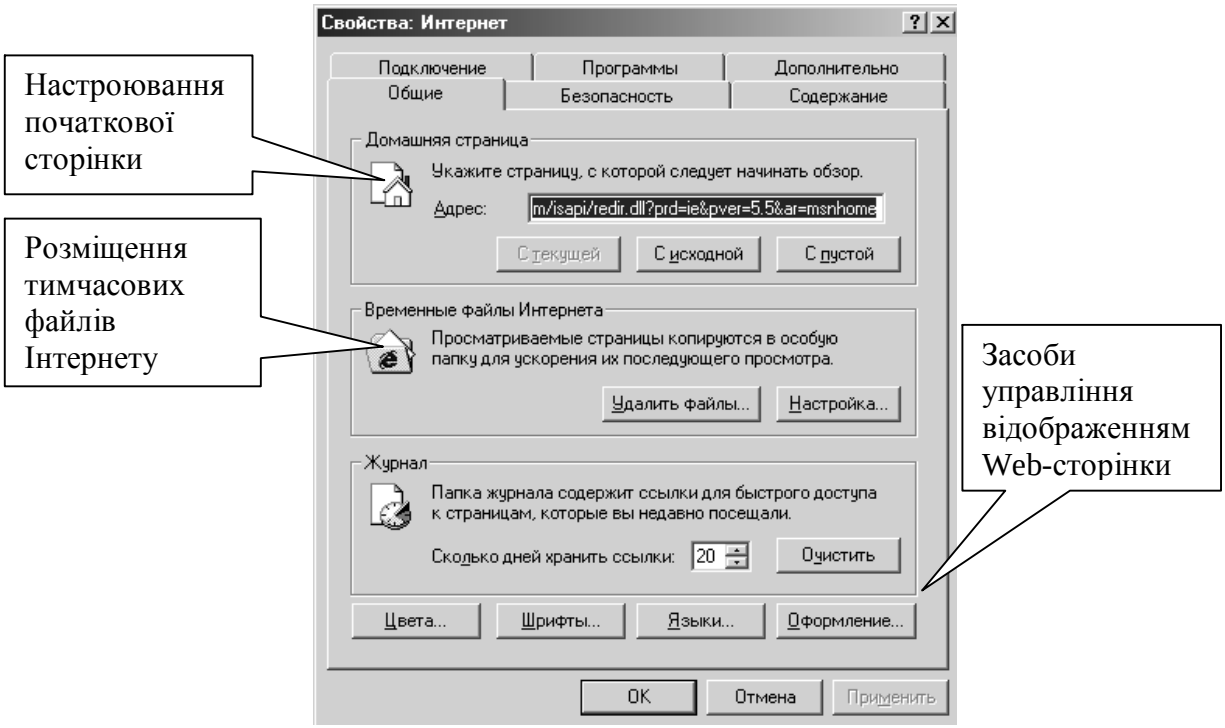



**Рис. 5.3.** Управління основними параметрами відображення Web-

Кнопка **Оформление** дозволяє примусове використання параметрів

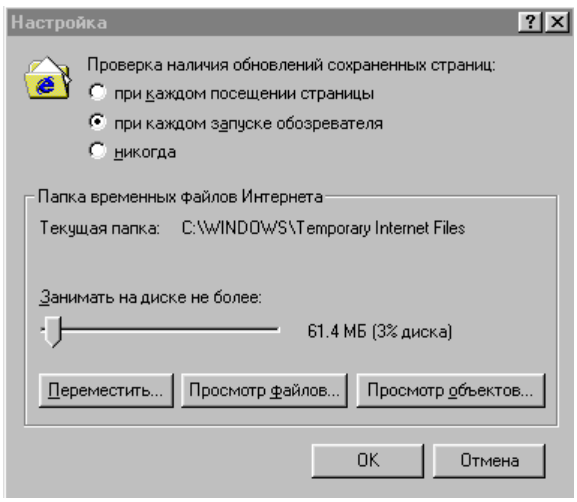



**Рис. 5.4**. Вікно **Настройка**

форматування, які задані у властивостях броузера. Це можна віднести до кольорів (прапорець **Не учитывать цвета, указанные на веб- страницах**), накреслення шрифтів (**Не учитывать шрифты, указанные на веб-страницах**) і розмір шрифтів (**Не учитывать шрифтов, указанные на веб-страницах**).

За допомогою закладки **Подключение** здійснюється настроювання властивостей з'єднання з Інтернетом, як і при використанні папки **Удаленный доступ к сети**. Також можна указати, яке з'єднання повинно використовуватися при роботі броузера. За допомогою перемикачів можна задати режим відмови від автоматичного підключення, стандартний режим підключення при відсутності з'єднання або режим використання тільки одного з'єднання.

За допомогою закладки **Программы** здійснюється вибір програм, що використовуються в Інтернет. Усі види програм, крім календаря, входять безпосередньо в дистрибутивний пакет Internet Explorer 5.0.

Закладка **Безопасность** пропонує засоби захисту від небезпечного змісту Web-документу. Вона дозволяє вказати Web-вузли, взаємодія з якими вважають небезпечним, і забороняє прийом з них інформації, котра може бути руйнувальною.

Елементи управління закладки **Содержание** слугують для обмеження доступу до вузлів з безпечним змістом і для управління використанням електронних сертифікатів.

На закладці **Дополнительно** зосереджені інші настройки, які дозволяють:

- додержувати конфіденціальність роботи за допомогою засобів шифрування, використовувати електронні сертифікати і своєчасно вилучати тимчасові файли;
- контролювати використання засобів мови Java;
- управляти зображенням мультимедійних об'єктів;
- використовувати додаткові налаштування оформлення;
- керувати режимом пошуку Web-сторінок, що мають потрібну інформацію.

*Прийом файлів із Інтернет*

Гіперпосилання на Web-сторінках вказують на документи різних типів. Коли броузер не може відображати файли певного типу (наприклад, використовуємо файли з розширенням .EXE,архіви .ZIP і інші), запускається процес завантаження цього файла на комп'ютер.

Програма Internet Explorer 5.0 виконує запуск майстра завантаження файла. На першому етапі роботи майстра можна відкрити файл і зберегти його на диску. Відкриття файла розуміє завантажування його в каталог тимчасових файлів і безпосередній запуск (коли це виконує мий файл) або відкриття за допомогою програми, яка запропонована для роботи з файлами цього типу. Зручніше і надійніше вибрати збереження файла на диску. В цьому випадку потрібно вибрати папку, в якій потрібно зберегти файл, і задати ім'я файла.

Хід завантаження файла відображається в спеціальному вікні. Шкала ходу роботи з'являється тільки в тому випадку, коли майстер управління завантаженням може отримати інформацію про повну довжину файла. Це можливо тільки тоді, коли файл завантажується безпосередньо з Web-вузла. При завантаженні файла з вузла FTP таки дані не наведені. За ходом завантажування можна також стежити по рядку заголовка вікна або, коли вікно згорнуте або закрите іншими вікнами, по напису на кнопці Панелі задач. Процес завантаження файла не заважає паралельному перегляду Web-сторінок або іншими операціям в Інтернеті.

Після закінчення завантаження вікно завантаження закривається автоматично, коли встановлений прапорець **Закрыть диалоговое окно после завершения загрузки**. В протилежному випадку після закінчення завантажування активізуються кнопки **Открыть** і **Открыть папку**, які відповідно дозволяють відкрити файл, що тільки що завантажений, або папку, яка його містить.

За допомогою кнопки **Отмена** можна в будь-який момент перервати завантаження файла. Після цієї операції або із за розриву з'єднання операцію завантаження треба повторити заново. В ОС Windows 98 нема засобів, які здатні відновити завантаження файлів, що перервані. Це можна зробити при використані спеціальних службових програм.

Файли, що доступні для завантаження користувачем, частіше за всього зберігаються на FTP-вузлах. Для доступу до FTP-вузла можна вказати його URL адресу на панелі **Адрес**. Броузер Internet Explorer 5.0 забезпечує за

замовчуванням *анонімне підключення* до FTP-вузла, при якому дозволені тільки перегляд каталогів і завантаження файлів. Коли анонімний доступ не дозволений, на екрані відображається діалогове вікно для введення імені й пароля, які потрібно знати.

Вікно FTP-вузла виглядає на екрані як звичайне вікно папки, але з використанням значка віддаленої папки. Для завантаження файла треба клацнути на його значку правою кнопкою миші і вибрати в контекстному меню команду **Копировать в папку**. Коли для цього каталогу FTP дозволені усі файлові операції, то з ним можна працювати як з вікном папки. Неможливо тільки зробити пряме перенесення файлів з одного вузла на інший. Щоб таке зробити, спочатку треба перенести файл в локальну папку комп'ютера, а потім відправити її звідти на інший FTP-вузол або в інший каталог того ж FTP-вузла.

Повідомлення поступають і відправляються через сервер, а тому програмі потрібно вказати інформацію про сервер, що використовується. Ця інформація зберігається у вигляді облікового запису.

## 5.4. Питання для самоконтролю

1. Що визначає термін World Wide Web?
2. Як розглядається середовище WWW?
3. Де зберігаються документи WWW?
4. Чи можна на Web-серверах розміщувати не окремий документ, а групу взаємопов'язаних документів?
5. Що таке Web-вузол (або Web-сайт)?
6. Що таке Web- видання або Web- публікація?
7. Що таке Web-сторінка?
8. Для чого використовується мова розмітки HTML?
9. Для чого використовують теги?
10. Від кого і через що отримують інформацію інтерактивні Web-сторінки і за допомогою яких спеціальних програм генерують запитану Web-сторінку?
11. Що таке Web-канали?

12. Що таке гіперпосилання?

13. Як здійснюється зв’язок між документами за допомогою гіпертекстових посилань?

14. Як можна організувати тематичну подорож по World Wide Web без використання адрес конкретних сторінок?

15. Яку форму використовує адресація документів?

16. На яку вказівку має адрес URL?

17. З яких частин складається адрес комп’ютера?

18. Яка служба перетворює адреси *URL* на цифрову форму *IP*-адреси?

19. Що забезпечує мова розмітки HTML?

Форматування і відображення документа на конкретному комп’ютері виконується спеціальною програмою — *броузером* (browser).

20. Яка спеціальна програма виконує форматування і відображення документа на конкретному комп’ютері?

21. Схема використання Інтернет через Internet Explorer.

22. Що забезпечує з боку Інтернет роботу служби World Wide Web?

23. Що забезпечує з боку користувача роботу служби в World Wide Web?

24. Які найбільш відомі броузери?

25. Які переваги є у броузера Microsoft Internet Explorer?

26. Як можна здійснити запуск броузера Internet Explorer?

27. Що з’являється на екрані, коли з’єднання з Інтернетом відсутнє?

28. Що з’являється на екрані при присутності з’єднання після запуску програми?

29. Що відображається в списку панелі Адрес?

30. Чи відображається адреса URL в рядку стану?

31. Яку команду застосовують, щоб відкрити нову сторінку, не закриваючи поточної?

32. Для чого використовується кнопка Назад?

33. Для чого використовується кнопка Вперед?

34. Для чого використовується кнопка Обновить?

35. Для чого використовується кнопка Остановить?

36. Для чого використовується кнопка Домой?

37. Що дозволяють команди меню Файл?

38. Що дозволяють команди меню Правка ?

39. Що дозволяють команди меню Вид ?

40. Що дозволяють команди меню Избранное ?

41. Що дозволяють команди меню Сервис ?

42. Яка команда використовується для відкриття нового вікна програми Internet Explorer?

43. Від яких факторів залежить настроювання броузера?

44. Що здійснюється за допомогою закладки Подключение?

45. Що здійснюється за допомогою закладки Программы ?

46. Які засоби захисту пропонує закладка Безопасность ?

47. Для чого служать елементи управління закладки Содержание ?

На закладці Дополнительно зосереджені інші настройки, які дозволяють:

48. Що можна здійснювати на закладці Дополнительно?

49. Яка програма виконує запуск майстра завантаження файла?

50. За допомогою якої кнопки можна в будь-який момент перервати завантаження файла?

# 6. КОМП'ЮТЕРНА БЕЗПЕКА

## 6.1. Поняття комп'ютерної безпеки

В обчислювальній техніці під поняттям комп'ютерної безпеки розуміють:

- надійність роботи комп'ютера ;
- збереження даних;

- захист інформації від внесення в неї змін несанкціонованими користувачами;
- збереження конфіденціальності даних документа при застосуванні електронної пошти.

Надійність роботи комп'ютерних систем в багатьох випадках забкзпечують застосованням власних засобів самозахисту.

## 6.2. Комп'ютерні віруси

*Комп'ютерний вірус* — це програмний код, що вбудований в іншу програму, або в документ, або у визначені області носія даних, і який призначений для виконання несанкціонованих дій на даному комп'ютері.

Основними типами комп'ютерних вірусів є:

- програмні віруси;
- завантажувальні віруси;
- макровіруси.

*Програмні віруси* — це блоки програмного коду, що цілеспрямовано вбудовуються всередину інших прикладних програм. При завантажені програми здійснюється запуск вірусного коду. Робота цього коду викликає сховані від користувача зміни у файловій системі жорстких дисків і/або змісті інших програм. Наприклад, вірусний код може відтворювати себе в тілі інших програм — цей процес має назву *розмноження*. Створивши достатню кількість копій, програмний код може зробити перехід до зруйнування: порушити роботу програм в операційній системі, вилучити інформацію, що зберігається на жорсткому диску. Цей процес має назву *вірусної атаки.*

Віруси можуть зробити запуск форматування жорстких дисків. В цьому випадку дані на жостком диску залишаються, але скористатися ними вже неможливо. Теоретично дані можна відтворити, але це дуже важко.

Апаратне та програмне забезпечення взаємопов'язані таким чином, що деяку хибу програмного забезпечення можна усунути тільки програмними засобами. Наприклад, базова система введення-виведення (BIOS) зберігає свої

мікропрограми команд в *флеш-пам'яті*. Можливість перезапису інформації в мікросхемі флеш-пам'яті використовують деякі програмні віруси для знищення даних BIOS. В цьому випадку для відновлення робото здатності комп'ютера потрібні або заміна мікросхеми, або її перепрограмування за допомогою спеціальних програмних засобів.

Програмні віруси заносяться на комп'ютер при запуску програм, що не перевірені, з дискети, компакт-диска тощо або з Інтернет. При звичайному копіюванні файлів, що мають віруси, комп'ютер заразитися не може. Усі дані, що поступають із Інтернету, повинні пройти обов'язкову перевірку на безпеку праці. Коли дані з'явилися із невідомого джерела, то їх слід знищити, не проглядаючи. Звичайний прийом розповсюдження "троянських" програм — це додатки до електронного листа з "пропозицією" викликати й запустити корисну програму.

*Завантажувальні віруси* — пошкоджують системні області магнітних носіїв (гнучких та жорстких дисків). Крім того, вони можуть тимчасово розташовуватися в оперативній пам'яті. Часто зараження здійснюється при завантажені комп'ютера з магнітного носія, системна область якого має завантажувальний вірус. Наприклад, при завантаженні комп'ютера з гнучкого диска вірус спочатку проникає в оперативну пам'ять, а потім в завантажувальний сектор жорстких дисків. Далі комп'ютер сам стає джерелом розповсюджування завантажувального вірусу.

*Макровіруси* — це віруси, що виконують пошкодження документів, які виконані в деяких прикладних програмах, що мають засоби для виконання *макрокоманд.* Наприклад, до таких документів відносяться документи текстового процесора з розширенням .DOC. Ураження здійснюється при відкриванні файла документа у вікні програми, коли в ній не відключена можливість виконання макрокоманд.

## 6.3. Методи захисту від комп'ютерних вірусів

Є три рівня захисту від комп'ютерних вірусів:

- запобігання надходженню вірусів;
- запобігання вірусної атаки, коли вірус вже знаходиться у комп'ютері;
- запобігання руйнуванню, коли атака вже здійснилася.

Є три метода реалізації захисту:

- програмні методи захисту;
- апаратні методи захисту;
- організаційні методи захисту.

Існує багато можливостей знищення важливих даних: віруси, апаратні та програмні збоїни, крадіжка, пожар, стихійні лиха. Треба зробити таким чином, щоб непередбачені лиха не привели до катастрофічних наслідків.

*Засоби ативірусного захисту*

Основним захистом інформації є резервне копіювання найважливіших даних. Треба також окремо зберігати всі реєстраційні й парольні дані для доступу до мережних послуг Інтернету. Звичайне місце збереження — це службовий щоденник в сейфу керівника підрозділу. Будуючи план засобів по резервному копіюванню інформації, треба передбачити окремо збереження даних поза комп'ютером. Резервні 2-3 копії конфіденціальних даних зберігаються на зовнішніх носіях, що зберігаються в сейфах окремих приміщень. Між копіями здійснюється *ротація.*

Відносно новим і достатньо надійним засобом збереження не конфіденціальних даних є їх збереження в WEB-папках на віддалених серверах в Інтернеті. Є служба безкоштовного виділення простору (до декількох Мбайт) для збереження даних користувача.

Допоміжними засобами захисту інформації є антивірусні програми і засоби апаратного захисту. Є достатньо багато програмних засобів ативірусного захисту, які мають наступні можливості:

- створення копії жорсткого диску на зовнішніх носіях (наприклад, на гнучких дисках);
- регулярна перевірка жорстких дисків в пошуках вірусів;
- контроль зміни розмірів та інших атрибутів файлів;

- контроль звернень до жорсткого диску.

*Захист інформації в Інтернеті*

Працюючи в Інтернеті, треба пам’ятати, що всі ваші дії фіксуються й протоколюються спеціальними програмними засобами і інформація про законні і незаконні дії обов’язково десь накопичується. Крім того, інформація вільно циркулює в обидва боки і в загальному випадку доступна усім користувачам інформаційного процесу. Інтернет є простором, в якому циркулюють договірні і фінансові зобов’язання. Ця електронна комерція потребує відповідного захисту від перегляду сторонньою особою. Одним із засобів захисту важливої інформації в відкритих системах є шифрування даних за допомогою криптографічних систем.

## 6.4. Питання для самоконтролю

1. Що розуміють під поняттям комп’ютерної безпеки?
2. Що таке комп’ютерний вірус?
3. Які основні типи комп’ютерних вірусів?
4. Що таке програмні віруси?
5. Що таке вірусна атака?
6. Звідкіля заносяться на комп’ютер програмні віруси?
7. Що поражають завантажувальні віруси?
8. Що поражають макровіруси?
9. Які три рівня є захисту від комп’ютерних вірусів?
10. Які є три метода реалізації захисту?
11. Які існують причини знищення важливих даних?
12. Який є основний захист інформації від її знищення?
13. Де треба зберігати всі реєстраційні й парольні дані для доступу до мережних послуг Інтернету?
14. Які є допоміжні засоби захисту інформації?
15. Які можливості мають програмні засоби ативірусного захисту?

# 7. ОСНОВНІ ПОЛОЖЕННЯ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ

## 7.1. Загальні уявлення про інформаційну безпеку

Основу мережної служби складають програми *сервер* і *клієнт*, які працюють за установленими правилами. Ці правила закріплені у протоколах мережних служб. Операції, що виходять за рамки протоколів, вважаються небезпечними, а коли вони ще несанкціоновані, то вважаються забороненими і складають адміністративне правопорушення або кримінальний злочин.

*Протоколи* — це стандарти, які діють багато років. За цей час інформаційні технології розвиваються і виникає розрив між можливостями технологій і стандартами мережі. Програмісти постійно працюють над розширенням можливостей взаємодій між серверними і клієнтськими програмами. Виникає протиріччя між бажаним і дозволеним. В цій момент і виникає загроза мережної безпеки. Будь-яке розширення можливостей клієнтської програми є відходом від стандартного протоколу і може бути загрозою для мережі.

Питання мережної безпеки знаходяться у тонкому балансі між тим, що хочеться собі дозволити в мережі, і тим, що це буде коштувати для збереження безпеки в мережі. Загальний принцип такий: *чим складність системи більше, тим важче забезпечити в ній контроль за належним рівнем безпеки*.

Стандартні мережні засоби, які встановлюються разом з ОС Windows 98, не забезпечують ніякого рівня безпеки. Ці засоби зручно використовувати в якості учбових або побутових, але, коли звертаємося до корпоративного підключення до Інтернет службових комп'ютерів, необхідно застосовувати спеціальні засоби: або забезпечити відключення комп'ютерів від будь-яких службових даних, або використовувати інші операційні системи (UNIX, Linux тощо).

## 7.2. Поняття "належного" рівня безпеки

Коли комп'ютер використовується в якості учбового або довідкового, то турботу про безпеку можна звести до регулярного обслуговування і захисту від комп'ютерних вірусів і інших небезпечних програм.

Коли комп'ютер зберігає конфіденціальні дані, то треба забезпечити достатній рівень безпеки. До конфіденціальних даних відносяться: державні таємниці, секретні дані, а також дані персонального обліку: паспортного, медичного, освітнього тощо. Спеціалісти, що займаються безпекою зберігання конфіденціальних даних на комп'ютері, можуть забезпечити належний рівень безпеки, але це не проста дія і вельми дорога.

Розглянемо учбовий заклад університетського типу. Його комп'ютери знаходяться в комп'ютерних класах, в бібліотеці, в адміністративних відділах. Коли комп'ютери комп'ютерних класів (лабораторій) підключені до Інтернету, то на них не повинні знаходитися конфіденціальні дані (адреси студентів, викладачів, успішність тощо). З іншого боку, подібні дані можуть бути на комп'ютерах адміністративного відділу, які треба підключати до Інтернету з відповідними засобами захисту інформації.

На державних підприємствах та комерційних структурах забезпеченням режиму безпеки займаються *системні адміністратори.* Вони повинні робити регулярний моніторинг мережі в пошуках відомостей про методи порушення режиму безпеки, установку перевірених апаратних і програмних засобів захисту, контроль над тим, які програми установлюються користувачами і як вони їх експлуатують. Адміністрація комерційних підприємств, які діють також в галузі туризму (готелів, тур фірм тощо), повинні вирішувати кожного разу, що їм вигідніше: найняти на роботу спеціаліста по захисту даних і встановити дешеві пристрої або виділити для роботи в мережі дешеві комп'ютери, на яких взагалі не будуть зберігатися ніякі конфіденціальні дані і безпекою яких можна знехтувати.

На режимних підприємствах питання підключення того чи іншого комп'ютера до мережі вирішується завжди індивідуально для кожного робочого місця. Забезпечення їх безпечного функціонування коштує на багато дорожче (коли взагалі це можливо), ніж застосування спеціальних засобів, які пройшли сертифікацію у спеціалістів в галузі захисту інформації.

## 7.3. Основні види порушень режиму мережної безпеки

*Загроза віддаленого адміністрування.* Під віддаленим адмініструванням розуміють несанкціоноване управління віддаленим комп'ютером. Віддалене адміністрування дозволяє брати чужий комп'ютер під своє управління. Це дозволяє копіювати і змінювати на ньому дані, встановлювати на ньому всілякі програми, в тому числі і небезпечну, використовувати чужий комп'ютер для здійснення кримінальних дій в мережі "від його імені".

*Загроза активного змісту.* Активний зміст — це активні об'єкти, що вбудовані у Web-сторінку. Активні об'єкти включають ще програмний код, який може бути агресивним. Агресивний програмний код (програма), який попадає на комп'ютер "жертви", має здібності вести себе як комп'ютерний вірус або як агентська програма. Наприклад, він може здійснювати руйнування даних, працювати як засіб віддаленого адміністрування або підготовлювати підставу для його встановлення.

*Загроза перехопленняу або заміни даних на шляхах транспортування.* Із застосуванням Інтернету в економіці загострюється загроза перехоплення або заміни даних на шляхах транспортування. Наприклад, розрахунок електронними платіжними засобами (картками платіжних систем) припускає відправлення покупцем конфіденціальних даних про свою картку продавцю. Коли ці дані будуть перехоплені, то немає гарантії, що цими даними не скористується злочинець. Крім того, через Інтернет передаються файли програм. Заміна цих файлів може привести до того, що замість потрібної програми клієнт одержить її аналог з "розширеними" можливостями, що буває дуже небезпечно.

*Загроза втручання у власне життя.* В наш час річний рекламний бюджет Інтернету складає декілька десятків мільярдів доларів США. Щоб збільшити свої прибутки від реклами багато компаній організують Web-вузли не стільки для того, щоб давати клієнтам мережні послуги, а скільки для того, щоб збирати про них персональні дані. Ці дані узагальнюються, класифікуються і постачаються рекламним і маркетинговим службам. Процес збору персональної

інформації автоматизований, не потребує практично ніяких витрат і дозволяє без відома клієнтів досліджувати їх переваги, смаки, прихильності.

*Загроза постачання неприйнятності змісту.* Не вся інформація, що поступає по Інтернету, вважається суспільно корисною. Є багато причин морально-етичного, релігійного, культурного й політичного характеру, коли людям може бути неприємна інформація, що постачається, і вони бажають від неї захисту.

В більшості країн світу Інтернет не розглядається як засіб масової інформації тому, що постачальник інформації не займається копіюванням, тиражуванням і розповсюдженням. Все це здійснює сам клієнт в момент використовування гіперпосилання. Тому звичайні закони про засоби масової інформації, що регламентують, що можна розповсюджувати, а що ні, в Інтернет поки що не працюють.

Правовий вакуум, що є в цьому питанні, з часом буде ліквідований, але багатьом клієнтам і сьогодні потрібні засоби захисту від постачання *неприйнятності змісту.* Функція фільтрації змісту, що постачається, покладається на броузер або на спеціальну програму, яка встановлена для цієї мети.

## 7.4. Захист від віддаленого адміністрування та троянських програм

Відомі два методи віддаленого адміністрування:

- установлення на комп'ютері "жертви" програми (аналог сервера), з якою злочинець може створювати віддалене адміністрування в той час, коли "жертва" знаходиться в мережі. Такі програми називають *троянськими*;
- вихід за рамки спілкування з клієнтською (серверною) програмою і безпосередньо впливати на операційну систему, щоб через неї досягти доступу до інших програм і даних. Програми, що використовуються для експлуатації вразливостей комп'ютерних систем, називаються *експлоитами*.

Для ураження комп'ютерною троянською програмою хтось повинен зробити запуск цієї програми на цьому комп'ютері.

Для захисту від віддаленого адміністрування слід дотримуватись наступних правил.

▼ Слід обмежити доступ чужих людей до мережного комп'ютера. Доступ закривається звичайними адміністративними засобами (фізичне обмеження доступу, парольний захист тощо).

Установка троянських програм на чужих комп'ютерах пов'язана з психологічним впливом на користувача. Найбільш часто практикується розповсюдження троянських програм у вигляді додатків до повідомлення електронної пошти. В тексті дається пояснення на скільки ця програма корисна.

▼ Ніколи не робити запуск нічого, що поступає разом з електронною поштою, незалежно від того, що написано в супровідному повідомлені. Це правило розповсюджується і на листи, що отримані від близьких, друзів і знайомих.

У злочинця є засоби підробляти адресу користувача таким чином, щоб вона виглядала, як лист від знайомого. Або лист під гаслом — "застосуй сам і передай товаришу".

▼ Ніколи не відправляй програми у вигляді додатка до повідомлення електронною поштою. Коли програма в мережі не має адресата, то ніхто не бере на себе відповідальність за її роботу.

Не має потреби застосовувати програми для Інтернету у вигляді компакт-дисків. В самому Інтернеті є багато тисяч корисних програм, які розповсюджуються безкоштовно або умовно-безкоштовно.

▼ Ніколи не встановлюйте на мережних службових комп'ютерах неперевіреного програмного забезпечення, яке розповсюджується у вигляді збірників на компакт-дисках.

## 7.5. Захист від помилок у програмному забезпечені

Атакам *програм-експлоитів* підлягають сервери. Стратегія злочинців реалізується за три етапи. На першому етапі вони з'ясовують склад програм і пристроїв в локальній мережі «жертви». На другому етапі розшукуються відомі помилки в програмах. На третьому етапі вони готують програми-експлоити для

експлуатації виявлених помилок. Боротьба із злочинцями здійснюється на всіх трьох етапах.

Адміністрація серверів контролює зовнішні звертання, метою яких є виявлення програмно-апаратної конфігурації сервера. Це дозволяє взяти порушника на облік до того, як він почне здійснювати реальну атаку.

В відповідальних випадках використовують спеціальні комп'ютери і програми, які виконують функції міжмережних екранів (щитів). Такі засоби також називають *брандмауерами* (firewall).Брандмауер не дозволяє переглядати склад програмного забезпечення на сервері чужим користувачам і не пропускає несанкціоновані дані і команди.

Адміністрація сервера повинна уважно стежити за публікаціями в мережі повідомлень про помилки, які виявлені в програмах, що використовуються. Період часу від виявлення помилок і їх знищенню найбільш відповідальний. Злочинець в цей час і може зробити шкоду.

## 7.6. Загроза та захист від отримання клієнтом коду

В склад Web-документів можуть входити вбудовані об'єкти. Такі об'єкти, як рисунки, відео кліпи і звукозаписи, є *пасивним змістом.* Це дані. Їх можна переглядати, копіювати, відтворювати, зберігати, редагувати на власному комп'ютері. Але для розширення можливостей броузера і сервера багато розробників Web-сторінок вбудовують в них *активні об'єкти* і *активні сценарії.* В цьому випадку мова йде про отримання клієнтом програмного коду в складі Web-сторінки, яка завантажується.

Активні об'єкти відрізняються від пасивних тим, що мають крім даних ще програмний код, який працює на комп'ютері клієнта. В загальному випадку сервер може постачати активні об'єкти будь якої природи — для цього потрібно тільки "умовити" користувача прийняти програмний код, який "розширює" можливості його броузера. Коли користувач не згоден застосовувати дані пропозиції, то залишається ще загроза від апплетів Java, елементів ActiveX, сценаріїв JavaScript і VBScript. На роботу з ними броузер вже налагоджений.

Треба налагоджувати броузер таким чином, щоб загроза була мінімальною. Коли на комп'ютері не зберігаються ніякі важливі дані або конфіденціальні відомості, то захист можна відключити і переглядати Web-сторінки у тому вигляді, як замислив це розробник. Коли загроза є, то треба відключити прийом апплетів Java, елементів ActiveX, сценаріїв JavaScript і VBScript. Є ще компромісний варіант — в кожному конкретному випадку запрошувати дозвіл на прийом того чи іншого активного об'єкта. В залежності від того, з яким вузлом установлено з'єднання (можна йому довіряти або ні), це питання кожний раз вирішується по різному.

В програмі Internet Explorer 5.0 засоби налагоджування захисту від активного змісту існують в діалоговому вікні **Правила безопасности (Сервер/ Свойства обозревателя/Безопасность/Другой)**.

## 7.7 Засоби захисту даних на шляхах транспортування

При використані Інтернету виникає потреба в захисті даних на шляхах транспортування (наприклад при ділових переговорах, грошових розрахунках за постачені товари і послуги тощо). Одночасно виникає потреба в посвідчуванні (ідентифікації) партнерів по зв'язку і підтвердження (аутентифікації) цілісності даних.

З виникненням електронної комерції в бізнесі і економіці щодня виникає велика кількість проблем. Їх кількість постійно зростає. Сьогодні в електронній комерції є захист і аутентифікація даних, а також ідентифікація партнерів за допомогою криптографічних методів.

## 7.8 Захист від втручання в особисте життя

*Збір відомостей про учасників роботи в Інтернеті.* На Web-сторінках є рекламні об'яви (*баннери*). При їх прийомі броузер встановлює зв'язки з їх власниками (з рекламною системою) і непомітно для себе реєструється в цієї системі. При переходах від однієї Web-сторінці до другої, ми створюємо свій психологічний портрет (*профіль*). По характеру Web-вузлів і Web-сторінок, що відвідуються, віддалена служба може визначити стать, літа, рівень освіти, вид занять, рівень добробуту і навіть характер хвороби людини, яка ніколи до них

не зверталася. Достатньо один раз зареєструватися де не будь під своїм іменем і прізвищем, і раніш зібрані абстрактні відомості приймають конкретній характер — таким чином створюються негласні персональні бази даних на учасників роботи в мережі.

Порівнюючи дані по різних людях або по одній і тій же людині, що отримані в різний час, системи створюють профілі не тільки окремих людей, але і колективів: сімей, робочих груп, підприємств. Отримані дані можуть бути використані легально і нелегально  Класифіковані бази даних є товаром: вони купуються і продаються, переходять із рук в руки і стають основою для діяльності багатьох організацій самого різного профілю.

*Джерела персональної інформації.* Найбільш простим джерелом інформації для збору відомостей про активність клієнтів Інтернет є маркери *cookie.* Принцип дії їх такий:

Застосовуючи протокол HTTP, броузер може відправити запит на поставку одного Web-ресурсу (документа HTML) і при цьому себе серверу не подати. Але іноді нам корисно при переході на нову Web-сторінку, щоб сервер нас впізнавав і продовжував почату раніш роботу. Для цієї мети розробники протоколу HTTP (понад 10 років позаду) створили механізм подання броузера серверу за допомогою так званих маркерів *cookie.*

Згідно протоколу HTTP сервер передає броузеру невеликий пакет даних, в якому зафіксована кодова інформація, що потрібна серверу для впізнання (ідентифікації) броузера і настройки на роботу з ним. Цей пакет тимчасово запам’ятовується в оперативній пам’ять комп’ютера і виконує роль маркера (мітки). Коли в процесі роботи в WWW броузер знову буде повертатися до того ж Web-вузла, то при звертанні до нього він подає маркер, що був вже прийнятий, і сервер розуміє з яким клієнтом він має справу.

Маркери можуть бути тимчасові і постійні. Тимчасовий маркер зберігається в оперативній пам’яті поки броузер працює. По закінченню роботи усі тимчасові маркери, що отримані від броузера, вилучаються. В більшості випадках, таких, як відвідування Інтернет - магазинів, можна було б

обмежитися тимчасовими маркерами, щоб магазин “не забував” свого клієнта, коли він переходить від одного відділу до іншого. Але сервери віддають перевагу постійним маркерам.

Коли броузер завершує свою роботу, постійні маркери, що накопичуються в оперативній пам’яті, переносяться на жорсткий диск у вигляді файлів *cookie.* Таким чином здійснюється маркерування жорсткого диску, а в загалі і всього комп’ютера клієнта. При наступних виходах в Інтернет при запуску броузера здійснюється читання маркерів *cookie*, що накопичувалися в пам’яті комп’ютера. Броузер подає їх серверам, які їх поставили.

Фізичної загрози маркери *cookie* комп’ютеру не представляють — це файли даних, які не містять програмного коду і тому не завдають шкоди комп’ютеру. Але вони є загрозою для втручання в особисте життя людини.

*Правовий режим маркерів cookie.* Є як мінімум два способи використання маркерів cookie для несанкціонованого збору відомостей про клієнтів: легальний і нелегальний.

Нелегальний механізм пов’язаний з тим, що сервер прочитає не тільки ті маркери, що він поставив, але і ті, що поставили інші сервери. Механізм був названий нелегальним тому, що експлуатація помилок комп’ютерних систем — це правопорушення.

Збір відомостей про клієнтів на основі маркерів *cookie* має дозвіл і в останній час широко ввійшов у практику. Він заснований на тому, що деяка служба (наприклад, рекламна) широко розповсюджує на Web-вузлах різного тематичного напрямку свої графічні об’єкти (наприклад, рекламні баннери). Користувачі цих Web-вузлів отримують разом з рекламними баннерами пов’язані з ними маркери *cookie* По цих маркерах можна відбудувати шляхи клієнтів по Web-вузлах Інтернету (Рис.7.1.).

*Інші джерела персональної інформації.* Підчас зв’язку по протоколу HTTP броузер повідомляє свою назву, номер версії, тип операційної системи комп’ютера клієнта і URL-адресу Web-сторінки, яку клієнт відвідував в останній раз.

Крім того, у серверів є можливість таємно отримувати адресу електронної пошти клієнта, що з правової точки зору викликає сумнів.

Ще одним джерелом персональної інформації є активні сценарії JavaScript (*Java-скрипти*).

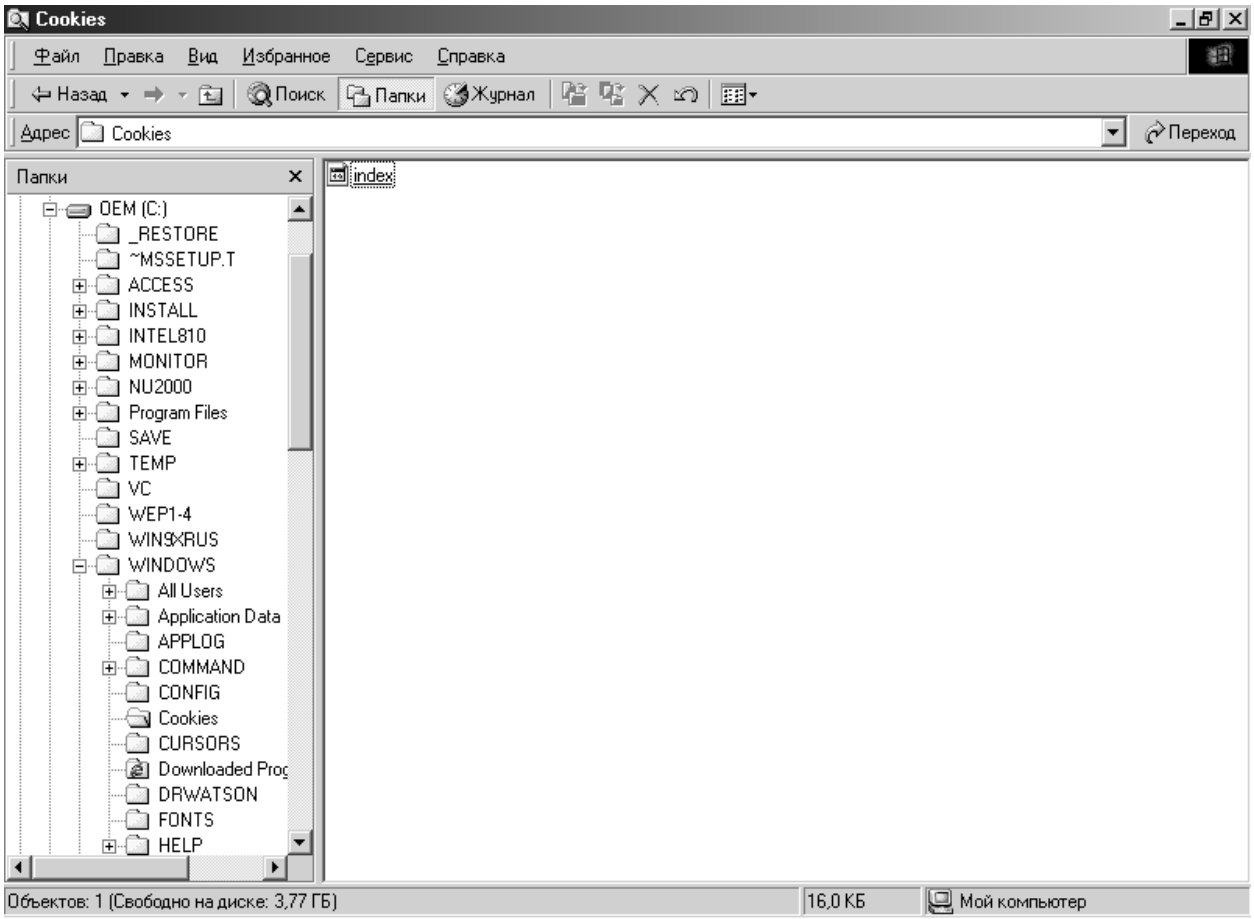



**Рис. 7.1.** Перегляд маркерів *cookie*, що зберігаються в папці *C:\Windows\Cookies*

## 7.9. Питання для самоконтролю

1. Що складає основу мережної служби?
2. Який час діють протоколи?
3. Який рівень безпеки забезпечують стандартні мережені засоби, які установлюються разом з ОС Windows 98?
4. Що відноситься до конфіденціальних даних?
5. Хто здійснює режим безпеки на державних підприємствах та комерційних структурах?
6. Яка є загроза від віддаленого адміністрування?
7. Що треба забезпечити, коли комп'ютер зберігає конфіденціальні дані?
8. Яка є загроза від активного змісту?
9. Яка є загроза від перехвату або заміни даних на шляхах транспортування?
10. Яка є загроза від постачання неприйнятності змісту?
11. Яка є загроза від втручання у власне життя через комп'ютер?
12. Які два методи є для ефектного захисту від віддаленого адміністрування?
13. На який пристрій йде атака програм-експлоитів?
14. За якими трьома етапами реалізується стратегія злочинців?
15. Які засоби називають *брандмауерами* (firewall)?
16. Які об'єкти є пасивним змістом? Що з ними можна робити?
17. Що таке активні об'єкти і активні сценарії?
18. В чому активні об'єкти відрізняються від пасивних?
19. В якому діалоговому вікні комп'ютера існують настройки захисту від активного змісту? Яка при цьому використовується команда?
20. Які є засоби захисту даних на шляхах транспортування даних?
21. Як здійснюється збір відомостей про учасників роботи в Інтернеті?
22. Які є джерела персональної інформації?
23. Для чого застосовуються маркери cookie?
24. Який принцип використання маркерів cookie?

# 8. ПОШУК ІНФОРМАЦІЇ

## 8.1. Вступ

Інтернет має три функції: *комунікаційну, інформаційну і управлінську*. Різні служби забезпечують різні функції. В рамці служби WWW є сервери, що використовують комунікаційні і управлінські функції. Основне призначення цієї служби — інформаційне. Коли нам потрібно розшукати якісь відомості, ми звертаємося за даними в першу чергу в інформаційний простір Web.

Інформаційний простір має в наступний час більше двох мільярдів Web-документів. Знайти серед них потрібну Web-сторінку або Web-документ це не проста задача. Для цього на службі WWW є пошукові сервери, які працюють безкоштовно. Економічну основу їх діяльності забезпечує високенний *коефіцієнт повернення* клієнтів.

Пошукова система — це спеціалізований Web-вузол. Користувач повідомляє пошуковій системі про зміст Web-сторінки, що шукається, а система надає йому список гіперпосилань на сторінки, що відповідають запиту. Є декілька моделей, на яких грунтується робота пошукових систем, але історично найбільш відомі дві моделі — це *пошукові каталоги* і *пошукові покажчики.*

## 8.2. Пошукові каталоги

Пошукові каталоги улаштовані за принципом тематичних каталогів великих бібліотек. Звернувшись до пошукового каталогу, ми знаходимо на його головній сторінці короткий список тематичних категорій. Кожний запис у списку категорій — це гіперпосилання. Клацнувши на ньому, відкриваємо нову сторінку пошукового каталогу, на якому дана тема визначена більш детально. Клацнувши на назві теми, відкриваємо сторінку зі списком розділів. Продовжуючи занурюватися в тему, можна дійти до списку конкретних Web-

сторінок і вибрати той ресурс, який більше всього підходить для розв'язання задачі.

Робота з пошуковими каталогами інтуітивно проста. Практично завжди вона завершується результативно. Пошукові каталоги створюються вручну. Висококваліфіковані редактори особисто переглядають інформаційний простір WWW, відбирають те, що за їх розумінням має суспільний інтерес, і заносять адресу в каталог. Найбільший пошуковий каталог світу є каталог Yahoo! (www.yahoo.com). Його обслуговують близька 150 редакторів. Його загальний обсяг Web-ресурсів складає близько мільйону Web-сторінок, що менше 0,1% від усіх ресурсів WWW.

Пошукові каталоги використовуються для первинного реферативного пошуку інформації за заданою темою. Пошукових каталогів в світі не дуже багато. Це зв'язано з великою трудомісткістю створення їх змісту і обслуговуванню, а також з браком кваліфікованих кадрів редакторів. Найбільший пошуковий каталог Росії — Атрус (atrus.aport.ru).

## 8.3. Пошукові покажчики

Основною проблемою пошукових каталогів є низький коефіцієнт охоплення ресурсів WWW. Щоб розв'язати цю проблему, роботу по пошуку автоматизують. При автоматизації якість посилань знижується, але зростає їх кількість. Автоматичну каталогізацію Web-ресурсів і задоволення запитів клієнтів виконують *пошукові покажчики.*

Основний принцип роботи пошукового покажчика є пошук Web-ресурсів за *ключовими словами.* Користувач задає опис ресурсу, який відшукується, за допомогою ключових слів, після чого дається завдання на пошук. Пошукова система аналізує дані, що зберігаються в її базі, і надає список Web-сторінок, що відповідають на запит. Разом з гіперпосиланнями надається коротке повідомлення про знайдені ресурси, на основі яких користувач може вибрати

потрібний йому ресурс (рис.8.1).Сьогодні в світі існує близька 10 тисяч пошукових покажчиків. Вершину списка займають близька двох десятків іноземних систем: Alta Vista (www.atavista.com), Exite (www.exite.com), Fast Search (www.alltheweb.com), Go/Infoseek (www.go/com), GoTo (www.goto.com), Google (www.google.com), HotBot (hotbot.lycos.com), Inktomi (www.inktomi.com), Lycos (www.lycos.com), Netscape Search (search.netscape.com), Norhern Light (www.northernlight.com), WebCrawler (www.webcrawler.com) і інші. В Росії також є великі пошукові покажчики: Апорт2000 (www.aport.ru), Яndex (www.yandex.ru) і Рэмблер (www.rambler.ru).

Різні пошукові покажчики можуть використовувати різні інформаційні технології для обробки запитів користувачів. Щоб ефективно виконувати пошук інформації в WWW, потрібно уявляти переваги і недоліки кожної із систем і в загальних рисах розуміти принципи їх роботи.

*Три етапи роботи пошукового покажчика.* Роботу пошукового покажчика умовно можна розділити на три етапи. Перші два етапи — підготовчі і непомітні для користувача, третій — це взаємодія з користувачем. Від кожного із етапів залежать функціональні можливості пошукової системи і ефективність роботи з нею.

*Збір первісної бази даних.* На першому етапі пошукова система займається скануванням інформаційного простору WWW. Для цього використовуються спеціальні агентські програми — *черви.* Черви — це дуже ефективні мало розмірні броузери. Їх задача складається з автоматизації розшуку в мережі Web-ресурсів і, коли цей ресурс системі невідомий, копіювати його у свою базу даних. В такий же спосіб здійснюється і оновлення раніше отриманих документів, але змінених за час після попереднього копіювання.

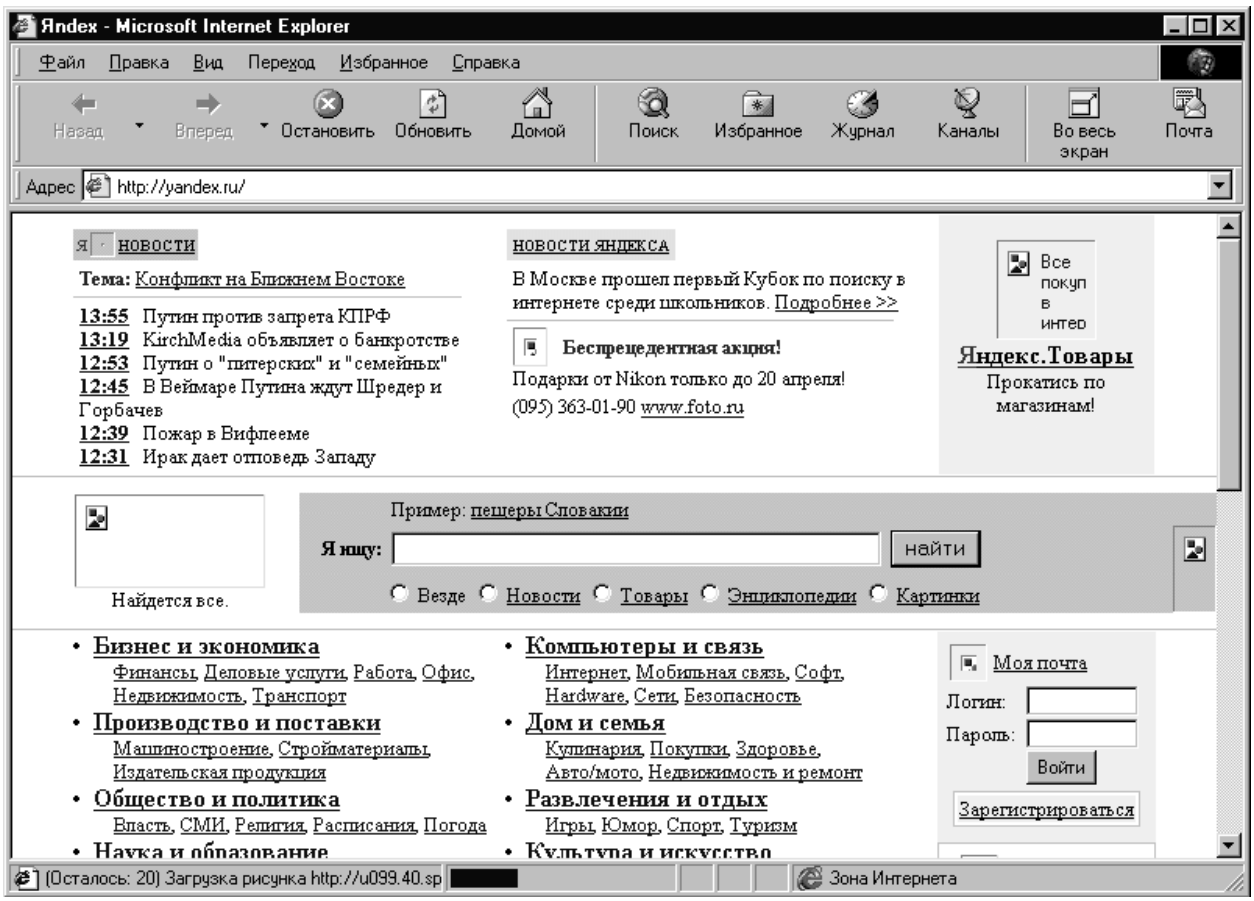



**Рис. 8.1.** Web-сторінка в період завантаження

Від роботи пошукового черва залежить змістовна частина пошукового покажчика. Кожна система використовує власну пошукову програму і зберігає у таємниці алгоритм її роботи від конкурентів.

*Індексація бази даних.* Зібрана база даних мережних Web-ресурсів — це ще недостатня основа для функціонування пошукової системи. З її допомогою можна обслуговувати запити клієнтів, але неможливо зробити це *швидко*. Пошук ключових слів в базі даних вельми довга операція. Тому зібрані бази даних проходять попередню обробку — *індексацію.* На етапі індексації створюються спеціалізовані документи — *пошукові покажчики.* Найпростіший тип пошукового покажчика має назву *зворотний файл.* Це словник, в який входять усі слова, що зустрічаються при перегляді Web-ресурсів. Напроти кожного слова надається список посилань, які вказують на розташування відповідних ресурсів в базі даних.

При отримані ключових слів від користувача перегляд пошукового покажчика здійснюється швидко, тому що вони попередньо були відсортовані за алфавітом. В результаті клієнт швидко отримує список посилань з цікавими для нього Web-ресурсами.

*Рафінування списку.* На третьому етапі створюється список посилань, який передається користувачу. Якість роботи пошукової системи залежить від технології, що використовувалася на цьому етапі.

*Рафінування* списку — це фільтрація і ранжирування результатів пошуку. Під фільтрацією розуміють відсів посилань, які не надаються користувачу. Потім йде перевірка наявності дублікатів. Посилання, що дублюються, перевантажують список користувача і затримують вибір потрібних ресурсів.

Ранжирування полягає в створенні спеціального порядку визначеного списку, в якому найбільш “корисні” (з точки зору пошукової системи) посилання подаються на початку списку, а не “корисні” — в кінці. Розуміння критерій “корисності” для клієнта того або іншого посилання може бути різним.

Тому різні пошукові системи, працюючи з однаковими ресурсами, надають різні результати пошуку.

## 8.4. Проблеми сучасних пошукових покажчиків

Великі пошукові покажчики світу в наш час знаходяться у критичному стані. У загальної кризи пошукових покажчиків є ряд об'єктивних причин. Всі вони пов'язані з суб'єктивними протиріччями, що виникли в процесі розвитку WWW. В різних країнах ці кризові явища визначаються по різному.

Основним протиріччям, яке визначає кризу, є різна динаміка зростання інформаційного простору Web і самих пошукових систем.

В 1994 році коефіцієнт охоплення Web-ресурсів досягав 50%, а самі Web-ресурси досягли 100 млн Web-сторінок, із яких десятки мільйонів були проіндексовані.

В 1999 році охоплення впало приблизно до 30%, а в 2000 році — до 20%. З кожним роком воно продовжує падати. Простір Web розвивається за рахунок мільйонів користувачів і декілька пошукових систем не в змозі його індексувати.

Друге протиріччя — чисто економічне. В 1997-1998 роках розвиток інформаційного простору Web досяг таких масштабів, що для його індексації потрібні були більш вагомі апаратні, програмні і кадрові ресурси. В той період інтереси інвесторів почали повертатися до інформаційних служб Інтернету. В результаті здійснилося акціонування деяких пошукових систем. Це вплинуло на характер їх роботи. Великі пошукові системи почали після 1997 року штучно гальмувати індексацію і зосереджуватися на боці комерційної діяльності. Багато з відомих раніше пошукових систем перетворилися в яскраві Web-портали, але з задачами наукового пошуку стали справлятися незадовільно.

З ростом WWW стали намічатися протиріччя, що зв'язані з інтересами клієнтів. Коли пошукова система видає багато посилань або дуже мало, то скористатися ними дуже важко. Клієнту потрібні "самі гарні" посилання, які він в змозі охопити. Це теж загальмувало роботи з індексації Web. Деякі пошукові

системи зовсім припинили займатися збором інформації і її аналізом. Замість цього вони переадресують запити клієнтів іншим пошуковим системам, які добре оснащені технічно, а самі працюють на третьому етапі — фільтрації і ранжируванні отриманих результатів. Наприклад, багато пошукових систем опираються на пошукову систему Inktomi (www.inktomi.com), яка виконує пошукові операції за запитами інших пошукових систем.

Кризові явища поки ще не торкнулися вітчизняних пошукових систем. Це зв'язано з тим, що наші пошукові системи працюють лише з інформаційним простором в декілька десятків мільйонів Web-сторінок, що приблизно відповідає ситуації 1993 року для країн Заходу. Це дає перевагу пошуковим системам і запас часу на декілька років, поки вони не зіткнуться з критичним явищем.

## 8.5. Новітні пошукові технології

*Автоматична каталогізація.* Протиріччя між розмірами досліджуваного і не досліджуваного Web-простору для пошукових каталогів ще гостріше, чим для пошукових покажчиків. Але тут є перспективний напрям розвитку. Він заснований на впровадженні SMART-технології автоматичної каталогізації. Найбільш перспективною є модель векторного інформаційного простору. Прикладом цього може бути експерт в юриспруденції, який складає словник для таких областей, як **Авторское право**, **Гражданское право**, **Уголовное право** і таке інше. Зробивши аналіз багатьох документів , які відносяться до цих наукових областей знань, він зуміє не тільки вказати характерні терміни і поняття, але і дати їм вагові оцінки. Зрозуміло, що слово "договір" має більшу вагу в документах цивільного права, чим кримінального права. Комбінуючи терміни і вагомі коефіцієнти, можна створювати багатомірні системи координат, в яких різні області знань описуються різними багатовимірними векторами.

Автоматично отримавши нову Web-сторінку, пошукова система може побудувати для неї математичний вектор, який грунтується на формальному

аналізі її змісту. Порівнюючи цей вектор з існуючими векторами для різних областей знань, пошукова система може без участі людини визначити, до якої категорії, теми і розділу відноситься той чи інший документ.

При такому підході не обов'язково зберігати копії всіх відомих Web-сторінок і їх пошукових покажчиків. Достатньо для кожного Web-документу зберігати лише його URL-адресу і число, що відповідає вектору. На сьогодні конкретні алгоритми SMART-технології не друкуються, тому що це ноу-хау. Але можна припустити, що вони працюють в пошукових системах реального часу, таких, як Alexa (www.alexa.com).

*Пошукові системи реального часу.* Розглянемо технологію пошуку інформації на прикладі пошукової служби Alexa (www.alexa.com). Для роботи з цією службою користувач повинен підключитися до її центрального серверу, отримати від нього і встановити на своєму комп'ютері клієнтську програму. Ця програма підключається до броузера і працює як додаткова панель в вікні Microsoft Internet Explorer або Netscape Navigator.

При кожному запуску броузера клієнтська програма установлює з'єднання зі своїм центральним сервером і далі працює з ним . Вона передає серверу копії усіх Web-сторінок, які відвідував користувач. Система завжди готова запропонувати користувачу список інших Web-сторінок, які близькі йому за тематикою. Вона підготовлює цей список на основі попереднього досвіду, що отриманий в роботі з іншими користувачами. Таким чином можна отримати рекомендації, котрі було б важко (а іноді і неможливо) розшукати в WWW традиційними пошуковими засобами.

## 8.6. Рекомендації стосовно прийомів ефективного пошуку

Для проведення реферативного пошуку, коли тема задана достатньо широко, рекомендується використовувати пошукові каталоги, такі, як Yahoo! (www.yahoo.com) або "Атрус" (atrus.aport.ru). Це дозволяє швидко встановити місце розташування основних джерел. При знайомстві з джерелами слід приділяти увагу понятійній базі. Знання основних понять і термінів дозволяє

перейти до пошуку в пошукових покажчиках з використанням ключових слів, що найбільш точно характеризують тему.

При наявності первинних відомостей з теми пошуку, документи можна розшукувати в пошукових покажчиках, використовуючи прийоми *простого, розширеного, контекстного і спеціального пошуку.*

Під *простим пошуком* розуміють пошук Web-ресурсів за одним або кількома ключовими словами. Недоліком цього пошуку є отримання великої кількості документів, серед яких важко знайти потрібний.

При використанні *розширеного пошуку* ключові поля зв'язані між собою операторами логічних відносин. За допомогою логічних відносин пошукове завдання формується таким чином, щоб більш точно обмежити область відбору. Наприклад, щоб відбір був за датою друку або за типом даних.

*Контекстний пошук* — це пошук за точною фразою. Він зручний для реферативного пошуку інформації, але має доступ не в усіх пошукових системах. Ця операція забирає багато часу при її виконанні.

*Спеціальний пошук* використовують для пошуку Web-сторінок, які мають посилання на задані адреси URL, а також містять задані дані в службових полях (наприклад, в полі заголовку).

## 8.7. Рекомендації по використанню пошукових систем

Рекомендується використовувати пошукову систему Nortern Light (www.norternlight.com) для проведення наукових пошуків (зокрема, з економіки і права). В ній можна знайти розділи каталожного типу, які називаються Special Editions. Додатково система надає платні послуги з поставляння актуальних наукових доповідей, які знаходяться в розділі Special Collection.

Найбільш великою пошуковою системою є Fast Search (www.alltheweb.com), яка швидко розвивається. На сьогодні вона досягла одного мільярда унікальних Web-сторінок.

Пошукова система Alta Vista (www.altavista.com) зорієнтована на комерційні рішення. В останній час вона також почала надавати послуги контекстного пошуку.

В Росії діють три потужні пошукові покажчики: «Апорт 2000» (www.aport.ru), «Рэмблер» (www.rambler.ru) і Яndex (www.yandex.ru). Всі вони працюють достатньо швидко. Систему «Апорт 2000» зручно використовувати в операціях простого пошуку. Система «Рэмблер» не тільки пошукова, але і виконує функції Web-порталу. Система Яndex використовується при формуванні складних пошукових завдань, оскільки вона має гнучку мову для розширеного пошуку.

## 8.8. Питання для самоконтролю

1. Які три функції має Інтернет?
2. Що є в межах служби WWW для використання комунікаційних і управлінських функцій?
3. Скільки Web-документів має інформаційний простір в наш час?
4. Що складає економічну основу пошукових серверів, які працюють безкоштовно?
5. Які найбільш відомі пошукові системи?
6. Що повідомляє користувач пошуковій системі і що надає йому пошукова система?
7. За яким принципом будуються пошукові каталоги?
8. Як в пошуковому каталозі знайти конкретну Web-сторінку?
9. Яким чином створюються пошукові каталоги?
10. Який пошуковий каталог вам знайомий? Назовіть його.
11. З чим зв’язана обмеженнісств кількості пошукових каталогів ?
12. Яка основна проблема пошукових каталогів?
13. Який основний принцип роботи пошукових покажчиків?
14. Скільки у світі пошукових покажчиків?

15. Що потрібно уявляти, щоб ефективно виконувати пошук інформації в WWW?

16. На які три етапи можна умовно розділити роботу пошукового покажчика?

17. На якому із етапів використовують спеціальні агентські програми — черви? В чому полягає їх робота?

18. Для чого використовується індексація бази даних?

19. Що таке зворотний файл?

20. Що передається користувачу на третьому етапі роботи пошукового покажчика?

21. Що таке рафінування списку?

22. Що таке ранжирування?

23. В якому стані знаходяться великі пошукові покажчики?

24. В чому полягає криза пошукових систем?

25. Що таке SMART-технологія?

26. Що таке математичний вектор Web-сторінки?

27. Що потрібно зберігати для кожної Web-сторінки при SMART-технології?

28. Пошукові системи реального часу?

29. Які пошукові системи рекомендуються для проведення реферативного пошуку? Чому слід приділяти найбільш увагу?

30. Що розуміють під простим пошуком? Які його недоліки?

31. Що розуміємо під використанням розширеного пошуку?

32. Як здійснюється контекстний пошук?

33. Спеціальний пошук.?

## 9. САМОСТІЙНА РОБОТА

Самостійна робота по вивченню запропонованого матеріалу лекцій полягає в досягненні можливості відповісти на запитання, що поданні у кінці кожної лекції (питання для самоконтролю).

Для контролю знань за визначеними темами для заочної форми навчання пропонуються варіанти  контрольних робіт у вигляді таблиць та самостійних робіт.

## 9.1. Контрольна робота № 1 за темами 1, 2, 3

В усіх варіантах контрольних робіт дайте письмові відповіді на питання

| Варіант | Питання параграфа 1.5. | Питання параграфа 2.2. | Питання параграфа 3.3. |
|---|---|---|---|
| 1 | 1, 19 | 1, 16,31 | 1 |
| 2 | 2, 20 | 2, 17,32 | 2 |
| 3 | 3, 21 | 3, 18,33 | 3 |
| 4 | 4, 22 | 4, 19,34 | 4 |
| 5 | 5, 23 | 5, 20,35 | 5 |
| 6 | 9, 26 | 6, 21,36 | 6 |
| 7 | 10, 27 | 7, 22,37 | 7 |
| 8 | 11, 28 | 8, 23,38 | 8 |
| 9 | 12, 29 | 9, 24,39 | 9 |
| 10 | 13, 30 | 10, 25,40 | 10 |
| 11 | 14, 31 | 11, 26,41 | 11 |
| 12 | 15, 32 | 12, 27,42 | 12 |
| 13 | 16, 33 | 13, 28,31 | 13 |
| 14 | 17, 34 | 14, 29,32 | 14 |
| 15 | 18, 35 | 15, 30,33 | 15 |

## 9.2. Контрольна робота № 2 за темами 4, 5, 6

В усіх варіантах контрольних робіт дайте письмові відповіді на питання

| Варіант | Питання параграфа 5.4. | Питання параграфа 4.5 | Питання параграфа 6.4. |
|---|---|---|---|
| 1 | 1, 19, 28, 45 | 1, 16 | 1 |
| 2 | 2, 20, 29, 46 | 2, 17 | 2 |
| 3 | 3, 21, 30, 47 | 3, 18 | 3 |
| 4 | 4, 22 ,31, 48 | 4, 19 | 4 |
| 5 | 5, 23. 32, 49 | 5, 20 | 5 |
| 6 | 9, 6, 33, 50 | 6, 21 | 6 |
| 7 | 10, 7, 34, 25 | 7, 22 | 7 |
| 8 | 11, 8, 35, 27 | 8, 23 | 8 |
| 9 | 12, 24, 36, 45 | 9, 26 | 9 |
| 10 | 25, 37, 44, 46 | 10, 27 | 10 |
| 11 | 14, 26, 38, 47 | 11, 28 | 11 |
| 12 | 27, 39, 43, 48 | 12, 29 | 12 |
| 13 | 16, 13, 40, 49 | 13, 30 | 13 |
| 14 | 17, 14, 41, 2 | 14, 31 | 14 |
| 15 | 18, 15, 42, 50 | 15, 32 | 15 |

## 9.3 Контрольна робота №3 за темами 7,8

| Варіант | Питання параграфа 7.9 | Питання параграфа 8.8 |
|---|---|---|
| 1 | 1, 16 | 1, 16 |
| 2 | 2, 17 | 2, 17 |
| 3 | 3, 18 | 3, 18 |
| 4 | 4, 19 | 4, 19 |
| 5 | 5, 20 | 5, 33 |
| 6 | 6, 21 | 6, 21 |
| 7 | 7, 22 | 7, 22 |
| 8 | 8, 23 | 8, 23 |
| 9 | 9, 16 | 9, 26 |
| 10 | 10, 12 | 10, 27 |
| 11 | 11, 20 | 11, 28 |
| 12 | 12, 21 | 12, 29 |
| 13 | 13, 22 | 13, 30 |
| 14 | 14, 17 | 14, 31 |
| 15 | 15, 24 | 15, 32 |

# 10. ПОРЯДОК РОБОТИ З ЕЛЕКТРОННОЮ ПОШТОЮ

## 10.1. Загальний огляд

Електронна пошта реалізується за допомогою двох програм і протоколу між ними. З боку Інтернету працює програма-сервер, а з боку користувача — програма-клієнт. Робота електронної пошти основана на двох прикладних протоколах. Один використовується для надсилання повідомлення, а інший — для отримання кореспонденції. Збоку Інтернету електронна пошта забезпечується двома серверами: сервером вихідних повідомлень і сервером вхідних повідомлень. Сервер вхідних повідомлень часто називають “поштовим ящиком”. Необхідність в наявності двох різних протоколів пов’язана з вимогами безпеки. Служба вихідних повідомлень основана на протоколі SMTP (Simple Mail Transfer Protocol —простий протокол передачі пошти) і з боку Інтернет забезпечується серверами SMTP.

Служба вхідних повідомлень електронної пошти виконує роль поштового відділення. Вона перевіряє права клієнта на основі його реєстраційного імені і пароля, після чого поставляє йому кореспонденцію, що накопичувалась в його “поштовому ящику”. Найбільш розповсюдженим протоколом цієї служби є протокол POP3 (Post Office Protocol 3 — протокол поштового відділення, версія 3).

## 10.2. Загальний порядок роботи з електронною поштою

При роботі з електронною поштою користувач повинен виконувати наступні дії:

1. Зареєструватися на сервері електронної пошти, який надає послуги як SMTP, так і POP3. Реєстрація означає *отримання облікового запису.* Клієнт отримує обліковий запис SMTP і обліковий запис POP3. Він їх отримує у тієї організації, яка надає загальний доступ до Інтернет (у сервіс-провайдера).

2. Установити на своєму комп'ютері *програму — клієнт* електронної пошти. Як правило, робота з вхідною і вихідною електронною поштою забезпечується однією програмою, але це необов'язково.

3. Налаштувати поштового клієнта на роботу з обліковими записами SMTP і POP3. При налаштуванні в програму вводяться реєстраційні дані користувача (реєстраційне ім'я і пароль), які отримані за договором з постачальником послуг електронної пошти. Це зручно, щоб при підключенні до серверу POP3 програма могла пред'явити серверу права автоматично, не вводячи їх кожний раз вручну.

4. Створити за допомогою клієнтського програмного забезпечення повідомлення, ввести адресу електронної пошти одержувача і надати команду надсилання.

5. Далі повідомлення транспортується по ланцюгу поштових серверів від SMTP-сервера відправника до POP3-сервера одержувача, де надходження повідомлень накопичуються до моменту, коли адресат встановить з'єднання з сервером.

6. В момент підключення до свого "поштового ящика" адресат може отримати:

- сповіщення про наявність поштових повідомлень;
- службову інформацію, яку отримано із полів заголовка повідомлень (ім'я відправника, дату відправлення, розмір і тему повідомлення тощо);
- повний текст повідомлення.

Що отримує адресат, залежить від того, як працює (налаштована) його клієнтська програма. Користувач може керувати об'ємом інформації, що транспортується на його комп'ютер. Це можливо, коли його клієнтська програма має необхідні функції.

## 10.3. Електрона пошта

Електронна пошта, що заснована на протоколах SMTP і POP3, називається E-mail. Поряд з нею, в останні роки була побудована інша система електронної

пошти Інтернету, що основана на службі WWW і яка називається Web-Mail. Це не самостійна служба, а сервіс, який реалізований засобами служби World Wide Web на базі протоколу HTTP. З боку Інтернету цей сервіс підтримується звичайними Web-серверами, а з боку клієнта — достатньо мати звичайний Web-броузер.

Web-Mail — це емуляція роботи старої самостійної служби E-Mail засобами Web-технологій, яка підтримує повний набір її функцій. Таким чином, з боку користувача різниця тільки у тому, які програми використовуються для роботи, і у тому, як виконується підключення до служби.

## 10.4. Загальний порядок роботи з електронною поштою

1. Обліковий запис електронної пошти Web-Mail створюється на Web-вузлі служби. Підключення до нього виконується переходом за URL-адресою за допомогою броузера. Реєстрація виконується шляхом заповнення HTML-форми на одній із Web-сторінок.

2. Підготовка і відсилання поштового повідомлення виконується шляхом заповнення полів Web-форми. Далі повідомлення передається за відповідною адресою, . Створені повідомлення фактично є документами HTML і переглядаються за допомогою будь-якого броузера. Як документи HTML, вони можуть мати шрифтове й кольорове форматування, фонове зображення, вбудовану графіку і інші об’єкти. Це оззначає, що у форматі HTML можна переслати повідомлення китайською, японською, корейською і іншими мовами та ієрогліфами. Коли адресат не має облікового запису на Web-сервері, а користується “звичайною” електронною поштою з обліковим записом на сервері POP3, то формат повідомлення автоматично перетворюється у звичайний текст. Сьогодні багато поштових клієнтів, що працюють з обліковими записами POP3, також мають змогу відтворювати повідомлення у форматі HTML.

3. Повідомлення, що отримані на адресу Web-Mail, зберігаються на сервері і доступні для перегляду за допомогою броузера як звичайні Web-

сторінки. Свою “поштову скриньку” на Web-сервері можна розглядати як віддалену Web-папку і взагалі не запам'ятовувати інформацію у комп’ютері.

## 10.5. Структура повідомлення електронної пошти

До окремих повідомлень електронної пошти прийнято відноситись, як до окремих записів бази даних. У цьому сенсі “поштова скринька” POP3 розглядається як віддалена база даних, а повідомлення, що прийнято на комп’ютер, як локальна база даних. Приймання і відсилання повідомлень еквівалентні операціям копіювання записів з однієї бази в іншу, а програма-клієнт для роботи з електронною поштою є засобом роботи з базами даних.

Таким чином, кожне повідомлення — це окремий запис бази даних. Як будь-який запис, воно має поля, з якими можна працювати окремо, незалежно від іншого повідомлення.

Повідомлення електронної пошти складається з двох великих розділів: заголовка повідомлення і “тіла” повідомлення. Заголовок має такі поля: *Кому* (ім’я адресата); *Копия* (в цьому полі даються імена адресатів, яким адресована копія даного письма); *Тема* (в цьому полі дається тема повідомлення) та інші. Серед полів заголовку є поле дати, а також поле, у яке приймається інформація про наявність у повідомлення вбудованих файлів. Цей механізм дозволяє пересилати разом з текстовими повідомленнями документи нетекстової природи, наприклад мультимедійні, архівні, програмні та інші файли.

У основному розділі (у “тілі” повідомлення) міститься змістовний текст повідомлення. Умовно в ньому можна виділити два компонента: звернення і підпис.

## 10.6. Функції і властивості поштових клієнтів

У світі розповсюджено декілька сотень програм-клієнтів електронної пошти, яки різняться апаратно-програмними засобами. Для Windows найбільш відомими поштовими клієнтами є системи:

- Microsoft Outlook (постачається у складі пакету Microsoft Office);
- Microsoft Outlook Express (постачається у складі операційних систем Microsoft Windows);
- Eudora (www.eudora.com);
- Pegasus Mail (www.pegasus.usa.com);
- The Bat! (www.ritlabs.com);

При аналізуванні поштових клієнтів можна виділити:

- базові функції (основні);
- додаткові функції (розширені);
- спеціальні функції.

Базові функції поштових клієнтів використовуються для виконання найпростіших операцій по відсиланню та прийманні повідомлень електронної пошти.

*Приймання повідомлень та автономний перегляд* — це основна функція усіх поштових клієнтів. При підключенні до серверу POP3 виконується автоматичне копіювання усіх повідомлень, що надійшли у базу даних поштового клієнта. Потім їх можна читати у автономному режимі, відключившись від мережі.

*Створення нових повідомлень* — це друга важлива функція поштового клієнта. Для її реалізації програма може мати вбудований текстовий редактор. Для поштових клієнтів, які працюють в операційних системах Windows, достатньо підтримки операцій з використанням буфера обміну і можливості вибору кодування тексту.

*Упорядкування повідомлень* виконується шляхом групування й сортування. Групування виконується по логічних папках. Кожна папка — це своєрідний фільтр. Так, наприклад, у папці *Входящие* відображаються тільки прийняті повідомлення, в папці *Отправленные* — відіслані повідомлення, а в папці

*Исходящие* — повідомлення, що підготовлені для відсилання, але ще не відіслані з якоїсь причини. Багато поштових клієнтів мають спеціальну папку *Черновики* для збереження повідомлень, що ще не готові для відсилання.

Сортування виконується для упорядкування повідомлень у папках. Звичайний порядок сортування — за датою, але його можна змінити, коли поштовий клієнт має відповідні функції.

*При автоматизації підготовки відповідних повідомлень* поштовий клієнт в змозі використовувати дані, що взяті із полів заголовку вхідного повідомлення. Це спрощує заповнення полів *Кому*, *Тема* тощо. Зручною є функція цитування вхідного повідомлення. Вона дозволяє використовувати при підготовці відповіді обрані фрагменти вхідного повідомлення і виділити їх особливим способом, зробивши їх відмінними від тексту відповіді.

*Операції з вбудованими файлами* можуть використовувати коди, що не входять до стандарту ASCII. Особливістю цього стандарту є використання кодів, що не перевищують значення 127. У файлах, що вбудовані у текст повідомлення, зустрічаються коди зі значеннями від 128 до 255. Для пересилання їх разом з повідомленням електронною поштою потрібно спеціальне перетворення їх у комбінацію символів з кодуванням від 0 до 127. Поштовий клієнт адресата повинен виконати зворотне перетворення.

Такий механізм пересилання довільних файлів має назву поштових вкладень. Відправник повідомлення повинен вказати розташування файлу, який був приєднаний. При цьому поштовий клієнт виконує відповідний запис в одному із полів заголовка повідомлення, де вказується ім’я файла і використаний метод кодування. Основними методами кодування є: MIME, BinHex і UUEncode. Більшість сучасних поштових клієнтів застосовують саме ці методи.

При прийманні повідомлення поштовий клієнт дозволяє витягнути поштове вкладення і зберегти його у потрібному місці у вигляді автономного файла.

Розглянемо додаткові функції поштових клієнтів.

*Підтримка множинності ідентифікаційних записів.* Коли з програмою працюють декілька користувачів, то деякі поштові клієнти дозволяють кожному

із них створити власний ідентифікаційний запис і отримати власний комплект логічних папок. Передбачається, що при підключенні до поштового сервера здійснюється завантаження тільки тих повідомлень електронної пошти, які адресовані власнику поточного ідентифікаційного запису.

*Підтримка множинності облікових записів.* Коли користувач має декілька "поштових скриньок", що відкриті на різних серверах, то деякі поштові клієнти дозволяють вибрати поточний обліковий запис і здійснювати перемикання між записами.

*Підтримка облікових записів WWW.* Коли поштовий клієнт підтримує роботу з обліковими записами Web-Mail, то його можна використати одночасно і для роботи із "звичайною" електронною поштою, що заснована на Web.

*Підтримка формату HTML.* Коли поштовий клієнт підтримує формат HTML, то це дає можливість готувати, відсилати, отримувати і переглядати повідомлення, у яких є елементи форматування, використовується шрифтове і кольорове оформлення, є вбудовані мультимедійні об'єкти.

*Підтримка адресної книги.* Стандартними функціями роботи з адресною книгою є:

автоматизоване створення контакту на основі даних, що отримані із полів заголовку повідомлення, що надійшло;

автоматизоване створення шаблону нового повідомлення після вибору потрібного контакту;

упорядкування списку контактів шляхом групування й сортування.

*Розширення функцій вбудованого текстового редактора.* Додатковими функціями можуть бути макрокоманди для введення звернень і підписів, а також засоби генерації підписів шляхом випадкового вибору із створеного зовнішнього текстового файлу.

*Функції оповіщення.* Ці функції виконуються вбудованою додатковою системою, що використовується для контролю облікових записів на серверах POP3 і Web-Mail. Як сигнал оповіщення може використовуватися звуковий або візуальний сигнал (діалогове вікно).

*Засоби керування "поштовою скринькою".* У випадках, коли щоденний обсяг повідомлень дуже великий, використовують поштовий сервер, який має функції попереднього перегляду полів заголовків до завантаження повідомлень із серверу. Це дозволяє відмовитися від завантаження непотрібної кореспонденції. Всі операції відбору і вилучення виконуються безпосередньо на сервері, що знижує навантаження на канали зв'язку і зменшує витрати на їх експлуатацію.

*Фільтрування повідомлень.* Аналіз полів заголовків можна виконувати автоматично за допомогою програмних фільтрів. Фільтрування застосовується для боротьби з негативними явищами, які мають назву спам. Спам — це розсилка інформації, що не затребувана. Перше звернення, як правило, спамом не вважається. Його використовують рекламні служби електронної торгівлі. За допомогою фільтрування можна автоматично групувати у заданих тематичних або персональних папках. Наприклад, аналізуючи зміст поля *Кому*, програма може розподіляти пошту у різні папки, що відносяться до різних користувачів.

*Підтримка "чорного" і "білого" списків.* Засоби фільтрування можуть працювати зі списками поштових адрес, що вже заготовлені. "Чорний" список адрес електронної пошти блокується і знищується безпосередньо на сервері, а "білий" список використовується для отримання обраних повідомлень. В Інтернет є служба, яка веде облік відомих спаммерів і рекламних служб, які порушують етикет електронної пошти. Поштовий клієнт може автоматично звіряти адреси відправників вхідних повідомлень з даними цих мережних служб і блокувати вхідну кореспонденцію на основі їх рекомендацій.

*Функції слідкування і контролю за виконанням робіт.* В організаціях, де обробляються тисячі звернень у рік, для цієї цілі використовують спеціальні системи керування діловодством. У малих і середніх організаціях у багатьох випадках вдається обійтися поштовим клієнтом, який має розширені функції для контролю за рухом вхідних повідомлень по робочих місцях виконавців.

*Функції резервування і архівування.* Ці функції виконуються при отримані дуже важливої інформації. У випадках виходу з ладу поштової програми, є можливість повної відбудови усіх повідомлень.

*Функції автоматичної генерації відповіді і переадресації.* Автоматична генерація відповіді на поштове повідомлення дозволяє зберегти етикет електронної пошти і дати відповідь тоді, коли вас немає на місці з будь -яких причин. Можна класифікувати вхідні повідомлення за іменами відправників або за темами повідомлень і генерувати різні автовідповіді на різні звернення. Функція автоматичної переадресації дозволяє створити тимчасову "поштову скриньку" на сервері однієї із служб Web-Mail і виконати пере адресацію на нього кореспонденції, яка надходить на облікові записи серверів POP3. Робота з тимчасовою "поштовою скринькою" виконується за допомогою звичайного броузера із будь- якого місця зв'язку. Користувач має змогу переглянути усю отриману електронну пошту на одному доступному Web-вузлі.

## 10.7. Етикет електронної пошти

При розгляданні етикету електронної пошти слід окремо розглядати листування службове і особисте. Під службовою розуміють листування між організаціями і приватними особами. Особисте листування — це листування між приватними особами.

Розглянемо етикет службового листування.

1. При підготовці повідомлення -обов'язкове заповнення усіх полів заголовку.
2. "Тіло" повідомлення повинно мати конкретний запит. При відсутності чіткого формулювання запиту відповідь не гарантується.
3. Запит не повинен вимагати від адресату використання додаткових засобів зв'язку, таких як відвідування деяких Web-вузлів, відсилання факсу або телефонування за вказаною адресою.
4. Термін відповіді на повідомлення — 24 години.
5. При неотриманні відповіді на запит слід перевірити заповнення полів, чітку вказівку, коли чекається відповідь, підпис автора повідомлення. У службовому листуванні повторні звернення виконуються не раніше тижня, а інакше повідомлення може розглядатися як спам.

6. Не слід відсилатити ніяких поштових вкладень за власною ініціативою, а тільки за домовленням.

7. При отримані електронною поштою незамовленого повідомлення з вкладеним файлом таке поштове вкладення треба вилучити, не відкриваючи і не читаючи.

8. Коли надсилання поштового вкладення погоджено обома адресатами, слід зробити погодження розміру файла, що надсилається. Допустимим за розміром вважається файл до 100 Кбайт.

Розглянемо етикет приватного листування.

1. При підготовці повідомлення обов'язково заповнення усіх полів заголовку.

2. Звернення до незнайомої особи, яка не надрукувала свою адресу в мережі або в засобах масової інформації, вважається непристойним.

3. Термін відповіді на звернення не регламентується.

4. При підготовці відповіді слід використовувати мову і кодування символів оригінального повідомлення.

5. У випадку неотримання відповіді на повідомлення, надісланого незнайомій особі, повторні звернення неприпустимі і розглядаються як спам.

6. При отримані повідомлення, яке не очікується, від незнайомого джерела не рекомендується робити відповідь на це повідомлення.

7. При отримані повідомлення, яке не очікується і має вкладені файли, його слід вилучити, не відкриваючи.

8. При потребі вислати вкладений файл слід наперед повідомити партнера окремим повідомленням, щоб цей файл не було вилучено.

*Погодження електронної пошти*

У службовому листуванні використовується та мова й метод кодування символів, які затверджені у даній організації, незалежно від того, якою мовою і з яким кодуванням отримано повідомлення від кореспондента. У випадках сумніву слід звернутися до системного адміністратора.

У приватному листуванні слід використовувати ту мову і той метод кодування, які використані у вхідному повідомлені.

При підготовці повідомлень електронною поштою використовують емотикони і абревіатуру.

Емотикони — це комбінація символів, які використовуються для передачі емоцій автора. Емотикони створюються за допомогою клавіатурних комбінацій символів. Вони використовуються для передачі простих і зрозумілих почуттів, а не для створення загадок партнеру.

Прикладами емотиконів можуть бути такі комбінації клавіш:

:-) — посмішка; :-/ — іронія; :-( — печаль.

Кількість можливих емотиконів вимірюється багатьма сотнями. Їх використання дозволяє оживити неформальне листування.

Абревіатура — це стійке загальноприйняте скорочення слів, що використовують для спрощення клавіатурного вводу. Із найбільш широко використовуваних абревіатур слід відмітити такі:

BTW — by the way — вчасно...

IMHO — in my humble opinion — по моєму скромному розумінню...

AFAIK — as far as I know — наскільки мені відомо...

FYEO — for Your eyes only — строго між нами...

FYI — for Your information — до Вашої відомості...

Абревіатури особливо широко використовуються у повідомленнях груп новин (телеконференцій). Їх використовують тоді, коли практика електронного спілкування вже склалася.

## 10.8. Безпека електронної пошти

З точки зору безпеки, при роботі з електронною поштою виділяють таки загрози і вразливість:

- втрата конфіденціальності інформації;
- відмова в обслугованні;
- зараження комп’ютерним вірусом;

- проникання на комп’ютер активного змісту.

Щоб не втратити конфіденціальності інформації у поштовому обміну використовують методи симетричної й несиметричної криптографії. За симетричною криптографією обидва адресати використовують однакове шифрувальне й дешиврувальне програмне забезпечення. Зашифрувавши повідомлення за допомогою визначеного ключа (паролю), відправник повідомляє цей ключ адресату, використовуючи альтернативні засоби зв’язку (наприклад, телефон). При несиметричному шифруванні відправник шифрує повідомлення за допомогою сертифікату (відкритим ключем) того, хто отримує повідомлення. Більшість сучасних поштових клієнтів роблять ці операції автоматично, “непомітно” як для відправника, так і для адресата.

Загроза “відмови в обслугованні” пов’язана з цілеспрямованим виведенням з ладу поштового сервера адресата (наприклад, при переповнені повідомленнями пам’яті сервера). Як протидію використовують поштові клієнти, що здібні аналізувати повідомлення, що надходять на сервер. Щоб “поштова скринька” не мала переповнення, не слід широко публікувати свою адресу електронної пошти. У крайньому випадку, коли адресу опублікувати потрібно, використовують обліковий запис в одній з безкоштовних служб Web-Mail і використовують його як тимчасовий. При відсиланні своєї адреси у мережу, слід пам’ятати, що є автоматичні програмні засоби, які переглядають файли будь-яких типів у пошуках адрес E-Mail, які у них можуть бути. Ці засоби шукають у документах символ “@”, тому його замінюють іншим символом, зрозумілим людиною, але не програмою. Наприклад:

замість: myname@abcd.com

використовують: myname#abcd.com

Є ще більш надійний метод, коли замість імені адресата використовується шаблон, наприклад NOSPAM:

замість: myname@abcd.com

використовують: nospammyname#abcd.com

Що “nospam” потрібно вилучити, адресат повинен здогадатися сам.

Через механізм електронної пошти можна отримати як “класичні” комп’ютерні віруси, так і особливі “поштові” віруси. Класичні віруси розповсюджуються у вигляді виконуючих файлів, які вкладені у повідомлення електронної пошти. Усі виконуючі файли дуже небезпечні, коли вони надходять і від знайомих людей. Слід пам’ятати, що електронна пошта не застосовується для обміну програмами. Найбільш корисні і перевірені службові програми повинні бути опубліковані в мережі. В цьому випадку у тексті листа достатньо привести гіперпосилання на URL-адресу, щоб дану програму можна було отримати.

Механізм роботи “поштових вірусів” заснований на експлуатації вразливостей, які є в окремих поштових програмах. Цим атакам піддаються найбільш часто користувачі стандартного програмного забезпечення (наприклад, користувачі програми Outlook Express). Достатньо тільки відкрити файл, щоб отримати “поштовий вірус”. Тому повідомлення з вкладеними файлами слід вилучати, не переглядаючи його.

## 10.9. Питання для самоконтролю

1. За допомогою чого реалізується електронна пошта?
2. Яка програма працює з боку Інтернет, а яка — з боку користувача?
3. Призначення прикладних протоколів, на основі яких заснована робота електронної пошти.
4. Призначення двох серверів, які забезпечують електронну пошту збоку Інтернет.
5. Яку має назву сервер вхідних повідомлень?
6. Яка є потреба в застосуванні двох різних протоколів в електронній пошті?
7. На якому протоколі заснована служба вихідних повідомлень?
8. Призначення служби вхідних повідомлень електронної пошти?
9. Які дії повинен виконувати користувач при роботі з електронною поштою?

10. Яку має назву електронна пошта, що заснована на протоколах SMTP і POP3?

11. Яку має назву електронна пошта, що заснована на службі WWW?

12. Як можна розглядати окреме повідомлення з точки зору бази даних?

13. Які поля має заголовок повідомлення?

14. Що міститься у “тілі” повідомлення?

15. Що можна виділити при розгляданні поштових клієнтів?

16. Призначення базових функцій поштових клієнтів.

17. Прийом повідомлень та автономний перегляд.

18. Створення нових повідомлень.

19. Упорядкування повідомлень.

20. Сортування повідомлень.

21. Які дані в змозі використовувати поштовий клієнт при автоматизації підготовки відповідних повідомлень?

22. Які коди можуть використовувати операції з вбудованими файлами?

23. Механізм пересилання довільних файлів.

24. Підтримка множинності ідентифікованих записів.

25. Підтримка множинності облікових записів.

26. Підтримка облікових записів WWW.

27. Підтримка формату HTML.

28. Підтримка адресної книги.

29. Розширення функцій вбудованого текстового редактора.

30. Функції оповіщення.

31. Засоби керування “поштовою скринькою”.

32. Фільтрація повідомлень.

33. Що таке спам?

34. Підтримка “чорного” і “білого” списків.

35. Функції слідкування і контролю за виконанням робіт.

36. Функції резервування і архівації.

37. Функції автоматичної генерації відповіді і переадресації.

38. Яка різниця між службовим і особистим (приватним) листуванням?

39. Етикет службового листування.

40. Етикет приватного листування.

41. Що таке емотикони? Навести приклади.

42. Що таке абревіатура? Навести приклади.

43. Які виділяють загрози і вразливості з точки зору безпеки при роботі з електронною поштою?

44. Методи симетричної криптографії.

45. Методи несиметричної криптографії.

46. З чим пов’язана загроза “відмови в обслуговані” поштового серверу адресату. Які є засоби зменшення цієї загрози.

47. Як можна отримати “класичні віруси” через механізм електронної пошти?

48. На чому заснований механізм “поштових вірусів”?

## 11. САМОСТІЙНА РОБОТА

Самостійна робота по вивченню матеріалу лекцій, що запропонований, полягає в можливості відповідях на запитання, що поданні у кінці кожної лекції (питання для самоконтролю).

Для контролю знань по визначеним темам для заочної формі навчання подані варіанти виконання контрольних робіт у вигляді таблиць та самостійних робіт.

## 11.1. Контрольна робота № 4 за темою 10

В усіх варіантах контрольних робіт дати письмові відповіді на питання

| Варіант | Питання параграфа 10.9. |
|---|---|
| 1 | 1,16,33,46 |
| 2 | 2,17,34,47 |
| 3 | 3,18.35,48 |
| 4 | 4,19,37,46 |
| 5 | 5,20,38,47 |
| 6 | 6,21,39,48 |
| 7 | 7,22,40,46 |
| 8 | 8,23,41,47 |
| 9 | 9,24,42,48 |
| 10 | 10,25,43,46 |
| 11 | 11,26,44,47 |
| 12 | 12,27,45,48 |
| 13 | 13,28,31,46 |
| 14 | 14,29,32,47 |
| 15 | 15,30,36,48 |

## 11.2. Самостійна робота на тему: Засоби автоматизації при роботі з електронною поштою

**Тема:** Засоби автоматизації при роботі з електронною поштою.

**Мета:** Навчитися використовувати вбудовані засоби фільтрації.

Теоретичні відомості:

Автоматизація роботи з електронною поштою заснована на використані вбудованих засобів фільтрації. Ці засоби дозволяють вести оброблення вхідних повідомлень безпосередньо на сервері, до їх завантаження на комп’ютер.

Обробка здійснюється на основі аналізу даних із полів заголовка й реалізації заданої тактики:

- вилучення повідомлень до їх копіювання на комп'ютер;
- копіювання повідомлень у задану папку;
- автоматична генерація заданої відповіді тощо.

Тактика роботи з вхідними повідомленнями задається настроюванням системи правил. Настроювання системи правил виконується спеціальним засобом, котре запускається командою **Сервис/Правила для сообщений/Почта**. У вікні **Правила для сообщений**, що відкрилося, є командні кнопки для створення нового правила, зміни правила, яке існує, копіювання правила.

Розглянемо порядок створення нового правила для роботи з електронною поштою. Натиснути по кнопці **Создать** у діалоговому вікні **Правила для сообщений**, щоб відкрити вікно **Создать правило для почты.** В цьому вікні є три панелі.

На першій панелі вибрати умову для реалізації цього правила, наприклад **Искать сообщения с вложением**. За цією умовою будуть відібрані усі повідомлення, що мають вкладені файли.

На другій панелі вибрати дії з відібраними повідомленнями. Наприклад, вони можуть бути помічені, переадресовані на іншу адресу, вилучені із серверу без завантаження на комп'ютер, відправлені у потрібну папку тощо.

На третій панелі формуються готові правила. У цих випадках, коли для реалізації правила потрібно уточнити параметри (наприклад імені папки, в яку слід перемістити отримане повідомлення), цей параметр зображається синім кольором з підкресленням (у вигляді гіперпосилання). При натисканні на гіперпосилані можна вибрати або настроїти цей параметр.

Самостійно дослідити прийоми створення правил для роботи з повідомленнями. Створити правила для завдання, яке вказане у таблиці, що розташоване нижче. Вписати у порожні поля описи правил, які взяти із нижній панелі діалогового

вікна **Создать правила для сообщений**. Правила, що настроюються, виділити підкресленням, як це зображено на зразку.

**Таблиця**

| Завдання | Опис правил на російській мові |
|---|---|
| Відібрати повідомлення, які мають поштові вкладення і розмір яких перевищує 100 Кбайт. Не копіювати ці повідомлення на локальний комп'ютер | Шукати повідомлення, розмір яких перевищує 100 Кбайт і шукати повідомлення з вкладеннями.<br>Не завантажувати з сервера |
| Відібрати повідомлення, які отримані з обліковим записом www.hotmail.com, і переадресувати їх на свій обліковий запис www.mail.ua | |
| Автоматично відповісти на усі вхідні повідомлення стандартним повідомленням з вказівкою, що адресат знаходиться у відпустці і відповість на отримане повідомлення пізніше | |

Звіт за самостійну роботу здати викладачу.

# РОЗДІЛ 2. WEB-ПРОГРАМУВАННЯ

## 12. СТВОРЕННЯ WEB- СТОРІНКИ І БАЗОВІ ВІДОМОСТІ ПРО HTML

### 12.1. Гіперпосилання

*Гіперпосилання* (hyperlinks) дозволяють здійснювати перехід до інших документів і об'єктів Windows. Гіперпосилання мають вигляд клітинки, яка містить підкреслений текст. Коли курсор миші розташувати поверх клітинки з гіперпосилання, він приймає форму руки з вказівним пальцем. При натисненні на гіперпосилання мишею відкривається зв'язаний документ. При переході між сторінками за допомогою гіперпосилань зберігається послідовність перегляду усіх сторінок. Коли комп'ютер має доступ до Internet, гіперпосилання на Web-вузли відкривають Internet Explorer і доступ до відповідного Web-вузла. Наприклад, інформація про програми MS Office міститься на вузлі **http://www.microsort.com**. Коли файл в **URL** (Uniform Resource Locator, універсальний ідентифікатор ресурсів) не вказаний, то відкривається сторінка по замовчуванню цього Web-вузла.

Щоб на робочому аркуші MS Excel створити гіперпосилання на інший документ MS Office, можна зробити наступне:

1. Відкрити робочий аркуш і документ, з яким треба зв'язати робочий аркуш. Зберегти робочий лист і аркуш у папках.

2. Активізуйте документ, з яким повинний бути зв'язаний робочий лист MS Excel, і виділити текст або елемент, на котрий буде вказувати гіперпосилання.

3. Виконайте команду **Правка/Копировать.**

4. Активізуйте документ MS Excel і виділити клітинку, котра повинна містити гіперпосилання.

5. Виконайте команду **Правка/Вставить как гиперссылку.**

6. Натиснути на будь-яку клітинку, крім тієї, у котрій повинно бути створено гіперпосилання. Вбудований текст буде підкреслений і покрашений іншим кольором по відношенню до тексту іншої частини документа.

Щоб змінити гіперпосилання треба натиснути на ньому правою кнопкою миші. Із контекстного меню вибрати команду **Гиперссылка**, а потім вибрати одну із наступних команд (дивися рис. 12.1.).

Щоб ввести свій текст поверх гіперпосилання треба:

1. За допомогою клавіш зі стрілками виділити клітинку, яка містить гіперпосилання. Натиснути клавішу [F2].

2. Ввести текст і натиснути клавішу [Enter]. Те, що буде введено, і стане текстом гіперпосилання.

3. Можна змінити кольори та шрифт гіперпосилання за допомогою команди **Формат/ Ячейки.**

Гіперпосилання можна створити за допомогою функції ГИПЕРССЫЛКА робочого аркуша.

Синтаксис:

ГИПЕРССЫЛКА( *адрес_документа; имя*)

- *адрес_документа* — це шлях та ім’я файла для документа, що відкривається. Аргумент подається як текст.
- *имя* — текст переходу або числове значення, яке знаходиться в клітинці. Ім’я виділяється синім кольором із лінією підкреслення. Коли аргумент відсутній, клітинка в якості тексту переходу відображає значення аргументу *адрес*.

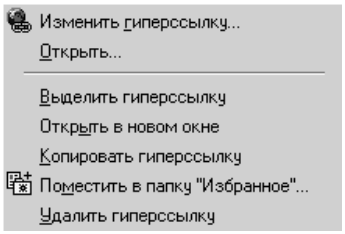



**Рис. 12. 1.** Команди

В якості значення параметрів функції ГИПЕРССЫЛКА може бути або текстовий вираз, або посилання на клітинку.

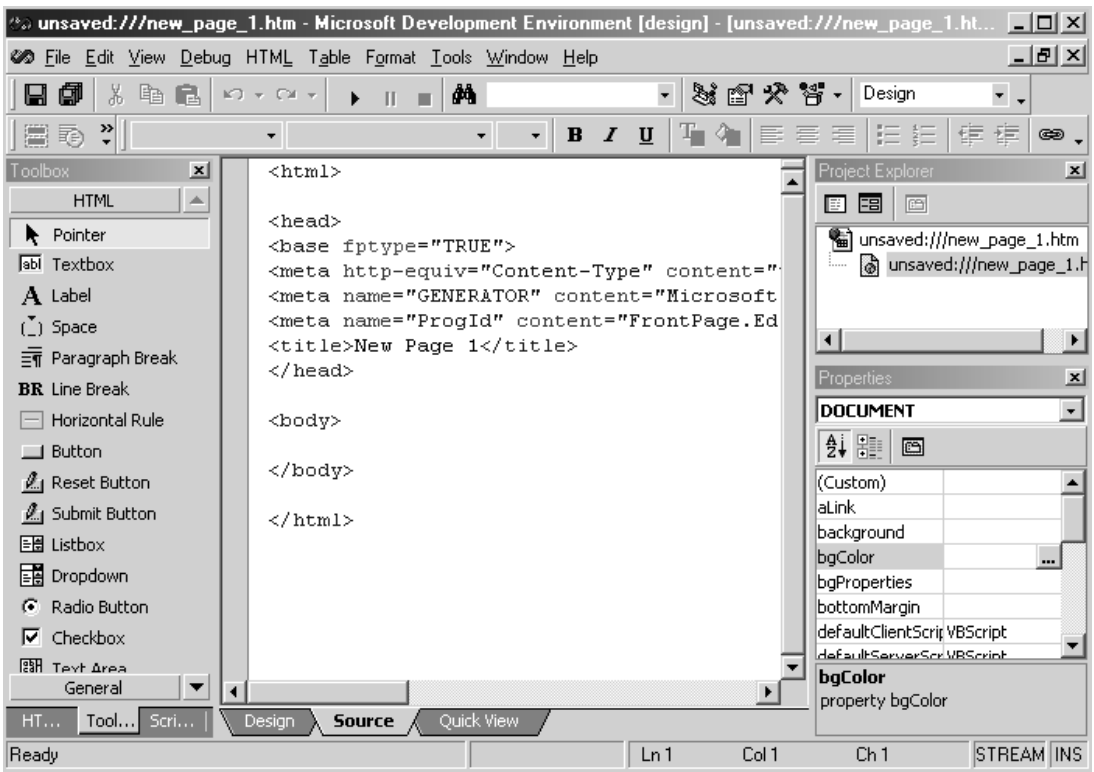



**Рис. 12. 2**. Microsoft Development Environment

## 12.2. Введення в HTML

Мова гіпертекстової розмітки HTML (Hyper Markup Language) — це мова, за допомогою якої створюються Web-сторінки. Для визначення зовнішнього вигляду документа мова розмітки дозволяє вставляти в текст документа спеціальні коди (дескриптори). Це можна зробити за допомогою текстового

редактора для Windows (наприклад, WordPad, Word, Notepad (програми Блокнот)).

Основні важливі характеристики мови розмітки є:

- HTML розроблена спеціально для створення WEB;
- у HTML включено гіпертекст;
- HTML підтримує мультимедіа.

*Гіпертекст* (Hypertext) — це текст, в якому є посилання на іншу Web-сторінку або документ.

*Гіперпосилання* (Hyperlinks) — це доповнення до додатків Windows, яке дозволяє зробити “стрибок ” до інших Web-сторінок або документів. Гіперпосилання дозволяє керувати переміщеннями по документу або документам.

*Дескриптор (тег)* — це основний елемент кодування, прийнятий у стандарті HTML. Дескриптори беруться у кутові дужки (<>).

*HTML* — це мова опису структури сторінки, яка дозволяє брати звичайний текст і форматувати його в абзаци, заголовки, списки ті інші структури. Ця мова пропонує засоби по створенню посилання на зв’язані сторінки, що дозволяє стрибати із одного документа в інший.

*HTML-файл* — це текстовий файл з розширенням htm (або html), у якому використані інструкції по форматуванню, що мають назву тегами (Tags) або дескрипторами.

Більшість тегів представлено у вигляді пар <TITLE> і </TITLE>, які починають і закінчують операцію.

Броузери ігнорують невидимі символи. *Броузер* (browser) — це додаток-клієнт, який використовує протокол HTTP для отримання й перегляду HTML-документів. *HTTP* (Hypertext transfer protocol) — протокол передачі HTML-документів.

## 12.3. Середовище розробки HTML - документів

Код HTML-документа можна складати в будь-якому текстовому редакторі. Після цього його слідує завантажити в Internet Explorer, щоб побачити його вигляд в якості Web-сторінки. Щоб переглянути HTML-код Web-сторінки, що завантажена, треба вибрати в Internet Explorer команду **Вид / В виде HTML.**

Для створення інтерактивної Web-сторінки використовується програма MS FrontPage, яка входить до MS Office і дозволяє розробнику створювати Web-сторінку візуальними засобами за допомогою миші і панелі інструментів. Програма MS FrontPage перекладе створену сторінку на мову HTML. Розробнику залишається тільки допрацювати цей код так, щоб документ мав змогу виконувати функції, що від нього потребує розробник Web-сторінки. Таким чином, програма MS FrontPage є середовищем для розробки Web-сторінки або Web-вузла.

Для створення HTML-документа зробіть запуск програми MS FrontPage. Виконайте команду **File/Save As**. На екрані з'явиться вікно **Save As.** В списку **Save as type** виберіть **Web Pages.** В полі **File Name** введіть ім'я файла, що зберігається, а в списку **Save in** вкажіть місцезнаходження файла, що зберігається. Натисніть кнопку **Save.** Таким чином, створений поки ще порожній HTML-документ.

Після цього можна зробити перехід до його конструювання. Вікно проекту складається з трьох закладок:

- Normal — в цьому вікні можна використовувати панелі інструментів Formatting і команди меню конструювання Web-сторінки;
- HTML — в цьому вікні відображається код Web-сторінки, що відповідає дизайну, який створений в вікні Normal. В цьому вікні розробник може редагувати, додавати в нього нові фрагменти для реалізації функцій;
- Preview File/in Browser. — це вікно попереднього перегляду Web-сторінки. Для перегляду всієї Web-сторінки в вікні броузера треба виконати команду **File/ Save.**

## 12.4. Універсальний ідентифікатор ресурсів URL

Адреса URL (Uniform Resource Locator) — універсальний ідентифікатор ресурсів — використовується в великій кількості в HTML-документах. URL — це адреса документа в Web. URL складається з чотирьох частин і має наступний вигляд:

*Протокол://ИмяСервера: НомерПорта/ ИмяФайла*

- *Протокол* — в більшості випадках він є HTTP (Hypertext transfer protocol, протокол передачі гіпертексту);
- *ИмяСервера* — англійське ім'я комп'ютера, на якому розташований документ або його IP-адреса (Internet Protocol);
- *НомерПорта* — часто не використовується і дорівнює 80;
- *ИмяФайла* — повне ім'я файла.

## 12.5. Теги

Умовно теги HTML можна розбити на частини:

- *Теги, які інформують інтерпретатор про те, що документ є HTML-документом, і теги коментарів.*

| **Тег** | **Опис** |
|---|---|
| <!-- … →, <COMMENT> | Текст, що розташований між парним тегом<COMMENT> або внутри тега <!-- текст коментарів --> є коментарієм. Коментарі не виводяться в вікні броузера. |
| <HTML> | Повідомлення броузера, що уся наступна за ним інформація є текстом, який закодований у відповідності з форматом HTML. Пара тегів <HTML> . . . </HTML> повинно охопити усе, за винятком, можливо, коментарів. |

- *Теги заголовка HTML-документа.*

| **Тег** | **Опис** |
|---|---|
| <HEAD> | Парний тег, який дозволяє створити заголовок документа. |
| <TITLE> | Парний тег, який служить для створення тексту, який буде |

| | |
|---|---|
| | відображатися в заголовку вікна броузера. |
| <BASE> | Задання повного URL документа. |
| <META> | Пропонує інформацію про документ для броузера, пошукові сервери і інші додатки. |

- *Теги тіла HTML-документа.*

Після заголовку в HTML-документі йде його тіло. Воно повинно починатися і закінчуватися парним тегом <BODY>. Між цими тегами розташовуються текст і теги документа.

Синтаксис:

<BODY

ALINK=*color*

BACKGROUND=*url*

BGCOLOR=*color*

BGPROPERTIES=FIXED

BOTTOMMARGIN=*pixels*

CLASS=c*lassname*

ID=*value*

LANG=*language*

LANGUAGE=JAVASCIPT | JSCRIPT | VBSCRIPT | VBS

LEFTMARGIN=*pixels*

LINK= *color*

RIGHTMARGIN= *pixels*

SCROLL=YES | NO

STYLE=*cssl –properties*

TEXT= *color*

TITLE=*string*

TOPMARGIN=*n*

VLINK= *color*

*event=script*>

Наведемо у вигляді таблиці базові кольорові константи.

| **Константа** | **Значення** | **Колір** | **Константа** | **Значення** | **Колір** |
|---|---|---|---|---|---|
| AQUA | #00FFFF | аквамарин | NAVY | #000080 | темно-синій |
| BLACK | #000000 | чорний | OLIVE | #808000 | оливковий |
| BLUE | #0000FF | голубий | PURPLE | #800080 | пурпурний |
| FUCHSIA | #FF00FF | бузковий | RED | #FF0000 | червоний |
| GRAY | #808080 | сірий | SILVER | #C0C0C0 | срібний |
| GREEN | #008000 | зелений | TEAL | #008080 | темно-зелений |
| LINE | #00FF00 | світло-зелений | WHITE | #FFFFFF | білий |
| MAROON | #800000 | каштановий | YELLOW | #FFFF00 | жовтий |

Наступний код може бути в якості шаблона HTML-документа.

```
< HTML>
<!—заголовок—>
<HEAD>
        <TITLE>
                Заголовок Web-сторінки
        </TITLE>
</HEAD>
<!—тіло—>
<BODY background=”Globe.gif” text=”#FFFF00”>
Тут розташований текст сторінки
</BODY>
< /HTML>
```

- *Теги форматування HTML-документа*

Теги, що використовуються при форматуванні HTML-документа.

| Тег | Опис |
|---|---|
| від <H1> до <H6> | Текст, що розташований між парою тегів <H*n*>…</H*n*>, відображається у вигляді заголовку *n*-го рівня. Шрифт першого рівня заголовку самий більший. Синтаксис:<br><P<br>ALIGN=CENTER \| LEFT \| RIGHT<br>CLASS=classname<br>ID=value<br>LANG=language<br>LANGUAGE=JAVASCRIPT \| JSCRIPT \| VBSCRIPT \| VBS<br>STYLE=cssl-properties<br>TITLE=text<br>event=script><br>Атрибут ALIGN задає допустимі значення для вирівнювання заголовків. Наприклад:<br><H1 align”center”><br>Заголовок першого рівня<br></H1> |
| <P> | Парний тег. Початок і кінець параграфа. Перед параграфом автоматично вставляється порожній рядок. Синтаксис:<br><H1 ALIGN=align-type> |
| <Center> | Парний тег. Центрує текст і малюнки. Наприклад:<br><CENTER><H1>Глава 1</H1></CENTER> |
| <BR> | Переведення рядка. Використовується для розриву рядка і відступу.<br><P><br>Don’t trouble troubles <BR><br>Title troubles trouble you. <BR><br>It only doubles trouble<BR><br>And also troubles you. <BR><br></P> |
| <HR> | Малює горизонтальну лінію. Синтаксис: |

| | |
|---|---|
| | <HR<br>ALIGN=CENTER \| LEFT \| RIGHT ‘тип вирівнювання лінії<br>CLASS=classname<br>COLOR=color ‘задає колір лінії у RGB-форматі<br>LANG=language<br>LANGUAGE=JAVASCRIPT \| JSCRIPT \| VBSCRIPT \| VBS<br>NOSHADE ‘лінія малюється без тіні<br>SIZE=n ‘товщина лінії у пікселях<br>STYLE=cssl-properties<br>TITLE=text<br>WIDHT=n ‘довжина лінії у пікселях або у відсотках за відношенням до довжини екрану<br>event=script><br>Наприклад,<br><HR aling=”right” color=”green” size=5 width=”25%”> |
| | Парні теги, що задають тип шрифту |
| <B> | напівжирний |
| <I> | курсивний |
| <U> | підкреслений |
| <STRIKE> | закреслений |
| <TT> | фіксованої довжини |
| <BIG> | більшого розміру |
| <SMALL> | меншого розміру |
| <SUB>,<SUP> | нижній і верхній індекс |
| <CITE> | використовується для виділення цитат |
| <CODE>,<PRE> | використовуються для виводу маленьких і найбільших фрагментів коду |
| <SAMP> | використовується для виділення декількох символів шрифтом фіксованої довжини<br>Наприклад:<br><B> Visual </B><I> Basic </I></BR><br>Команда <SAMP>Unload</SAMP> |

| | |
|---|---|
| <ADDRESS> | Парний тег. Використовується для ідентифікації автора і дати останнього оновлення Web-сторінки. Розташовується на початку або в кінці документа.<br>Наприклад:<br><ADDRESS><br>Created by garnaev </BR><br>Last Modified on 17 July 2000<br></ADDRESS> |
| <BLOCKQUOTE> | Парний тег. Використовується для виділення цитат. Він форматується як абзац із відступом від лівого і правого краю. |
| <FONT> | Парний тег. Задає розмір і колір шрифту. Синтаксис:<br><FONT<br>CLASS=classname<br>COLOR=color<br>FACE=font ‘ім’я шрифту<br>ID=value<br>LANG=language<br>LANGUAGE=JAVASCRIPT \| JSCRIPT \| VBSCRIPT \| VBS<br>SIZE=n ‘розмір шрифту<br>STYLE=cssl-properties<br>TITLE=text<br>event=script><br>Приклад виводу напівжирним курсивним шрифтом червоного кольору найбільшого розміру слово Test<br><B><I><FONT size=7 color=”red”>Test</Font></I></B> |
| <BASEFONT> | Задає стандартний тип шрифту для документу, а також його розмір та колір |

- *Тег зображення в HTML-документі.*

Зображення в документ вставляється за допомогою тега <IMG>.

<IMG

ALIGN=ABSBOTTOM | ABSMIDDLE | BASELINE | BOTTOM | LEFT | MIDDLE| RIGHT | TEXTTOP | TOP

ALT=text

BORDER=n

CLASS=classname

DATAFLD=colname

DATASRC=#ID

DYNSRC=url

HEIGHT=n

HSPACE=n

ID=value

ISMAP

LANG=language

LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS

LOOP=n

LOWSRC=url

NAME=name

SRC=url

STYLE=cssl –properties

TITLE=text

USEMAP=url

VSPACE=n

WIDTH=n

even=script>

Наведемо опис атрибутів, що найчастіше використовуються.

- ALIGN — задає вирівнювання зображення або оточуючого його тексту. Дозволяє створювати ефект оточення зображення текстом.
- ALT — текст альтернативного опису зображення, який буде з’являтися замість зображення, коли не встановлено налаштування **Отображать рисунки.**
- BORDER — ціле число, що задає товщину рамки зображення**.**
- CONTROLS — відображає керуючі елементи при демонстрації відеокліпів.
- DYNSRC — **URL** -відеокліпа.
- HEIGHT і WIDTH —цілі числа, що задають висоту і довжину зображення. Зображення автоматично заповнює усю відведену для нього область.

- HSPACE і VSPACE — цілі числа, що задають товщину бокових сторін, верхньої і нижньої частини рамки зображення. Подубни BORDER, але рамка в цьому випадку невидима.
- ISMAP — вказує, що відображення є картою образа, який реагує на натиснення кнопки миші.
- LOOP — ціле число, що задає число повторень відеокліпа. Допустиме значення –1 або INFINITE (нескінчене число разів).
- SRC — **URL** –відображення.
- START — задає подію, при якій починається програвання відеокліпа. Допустимі значення: FILEOPEN (при відкриванні файла), MOUSEOVER (при переміщенні вказівника миші над зображенням).
- USEMAP —фрагмент ідентифікатора для сайта клієнта з картою образу.

Приклад, як можна вставити графічне зображення і програти відеокліп:

<IMG align=MIDDLE src=”Globe.gif” width=100 height=100> Глобус

<IMG dynsrc=”Cineapk.avi”>

- *Тег вставки гіперпосилання і закладки*

Як закладка, так і вказівник гіперпосилання створюються за допомогою парного тега <A> (А від англ.. anchor). Синтаксис:

<A

ACCESSKEY=key

CLASS=classname

DATAFLD=colname

DATASRC=#ID

HREF=url

ID=value

LANG=language

LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS

METHODS=http-method

NAME=name

REL=”stylesheet”

REV=”stylesheet”

STYLE=cssl-properties

TABINDEX=n

TARGET=window_name | _blank | _parent | _self | _top

TITLE=text

URN=urn

Event=script>

Наведемо опис атрибутів, що найчастіше використовуються.

ˆ HREF — URL-адрес. Синтаксис: HREF=”type URL ”. Значеннями параметра type можуть бути:

- http: — для переходу на іншу Web-сторінку або закладку на іншій сторінці;
- file: — для завантаження і відкриття файла;
- mailto: — для відправки повідомлення електронною поштою за визначеною адресою.

ˆ NAME — ім’я мітки.

ˆ TARGET —визначає місце завантаження документу, на який вказує гіперпосилання. Можна задати або ім’я вікна, або фрейма, або одне із наступних спеціальних значень:

- _blank — відкривається нове вікно браузера, яке застосовується для виводу документа;
- _parent — завантаження документа виконується у батьківський фрейм поточного вікна ;
- _self — завантаження документа виконується у поточне вікно або фрейм;
- _top — завантаження документа виконується в усі вікна браузера, а не в фрейм

Тег <A> дозволяє відсилати також повідомлення за визначеною адресою електронною поштою. В цьому випадку атрибут HREF тега <A> має синтаксис HREF=”mailto:AddressE-Mail”. Другий приклад демонструє техніку використання гіперпосилання при відсиланні повідомлення електронною поштою.

<HTML>

<BODY bgcolor=”aqua”>

<H2>Visual Basic for Applications</H2>

<HR>

<B>Please e-mail questions or comments about this book to </B>

<A href=mailto:garnaev@somewhere.ru>A.Garnaev</A>

</ BODY>

</ HTML>

*Спеціальні символи.*

В HTML для зображення резервованих символів у коді використовуються спеціальні послідовності символів.(escape-послідовності):

| **Послідовність** | **Символ** | **Послідовність** | **Символ** |
|---|---|---|---|
| &lt; | < | &amp; | & |
| &gt; | > | &quot; | “ |
| &copy; | Знак авторського права © |   | Нерозривний пробіл |
| &red; | Знак резервованої торгової марки ® | | |

Для зображення символів <A> використовується вираз <A>

*Списки*

Списками є маркований, упорядкований та список визначень. Розглянемо перші два із них.

| **Тег** | **Опис** |
|---|---|
| <LI> | Парний тег. Визначає елемент у маркованому та упорядкованому списку. Використовується разом з парними тегами <UL> і <OL>. Синтаксис:<br><LI<br>CLASS=classname<br>ID=value<br>LANG=language<br>LANGUAGE=JAVASCRIPT \| JSCRIPT \| VBSCRIPT \| VBS<br>STYLE=cssl-properties<br>TITLE=text<br>TYPE=1 \| A \| i \| I<br>VALUE=value<br>event=script> |

| | |
|---|---|
| | Опис атрибутів, що найчастіше використовуються.<br>TYPE — задає тип нумерації упорядкованого списку і тип маркера маркованого списку. Допустимі значення: A (прописні букви), a (рядкові букви), I (прописні римські числа), i (рядкові римські числа), 1 (числа.);<br>VALUE — початковий номер. |
| <UL>, <OL> | Текст, що розміщується між парою тегів <UL>... </UL> створює маркований список, а між <OL>...</OL> — нумерований список. Елементи списку розміщуються між <LI>...</LI>.<br>Синтаксис цих тегів такий як і синтаксис парного тега <LI>.<br>Допустимо вкладення різнотипних та однотипних списків. |
| <DL> | Парний тег <DL> задає список визначень, що представляють собою текст у формі словникової статті |
| <DT> | Як правило парний тег. Відмічає визначений термін словника |
| <DD> | Як правило парний тег. Відмічає визначення терміну, що заданий тегом <DT> |

Розглянемо роботу зі списками на наступному прикладі, у якому створюються різноманітні вкладені списки.

```
<HTML>
<HEAD><TITLE>Приклад списків</ TITLE ></ HEAD>
<BODY>
<UL>
    <LI> Зима
        <OL>
            <LI type=”i”>Грудень</LI>
            <LI type=”i”>Січень</LI>
            <LI type=”i”>Лютий</LI>
        </OL>
    </LI>
    <LI>Весна
        <OL>
            <LI value=”4” type=”l”>Березень</LI>
```

```
                <LI value=”5” type=”l”>Квітень</LI>
                <LI value=”6” type=”l”>Травень</LI>
            </OL>
        </LI>
        <LI>Лето
            <OL>
                <LI value=”7” type=”l”>Червень</LI>
                <LI value=”8” type=”l”>Липень</LI>
                <LI value=”9” type=”l”>Серпень</LI>
            </OL>
        </LI>
        <LI>Осінь
            <OL>
                <LI value=”10” type=”a”>Вересень</LI>
                <LI value=”11” type=”a”>Жовтень</LI>
                <LI value=”12” type=”a”>Листопад</LI>
            </OL>
        </LI>
</UL>
</BODY>
</HTML>
```

Розглянемо роботу зі списком визначень, у якому створюється словник тегів HTML:

```
<HTML>
<HEAD><TITLE>Список визначень</ TITLE ></ HEAD>
<BODY>
<DL>
```

<DT>A

<DL>

<DT><A>

<DD>Задає закладки і гіперпосилання в документі

<DT><ADDRESS>

<DD>Використовується для ідентифікації автора документа

<DT>B

<DL>

<DT><B>

<DD>Напівжирний шрифт <DT><BASE>

<DD>Задає базовий URL для відносних URL у документі

</DL>

</DL>

</BODY>

</HTML>

*Таблиці*

Таблиці створюються за допомогою подвійного тега <TABLE>. Всередині цього тега розташовуються такі парні теги.

| **Тег** | **Опис** |
|---|---|
| <TR> | Рядок таблиці. Таблиця в HTML вводиться рядками, а кожний рядок складається із клітинок. |
| <TH> | Клітинка заголовку. У кожній клітинці заголовку текст виділений напівжирним шрифтом і вирівняний по центру. |
| <TD> | Клітинка таблиці. Клітинкою може бути об’єкт від однієї порожньої клитинки до цілої таблиці. |
| <CAPTION> | Заголовок таблиці. |

Синтаксис тега <TABLE>:

<TABLE

ALIGN=CENTER | LEFT | RIGHT

BACKGROUND=url

BGCOLOR=color

BORDER=n

BORDERCOLOR=color

BORDERCOLORDARK= color

BORDERCOLORLIGHT=color

CELLPADDING=n

CELLSPACING=n

CLASS=classname

COLS=n

DATAPAGESIZE=n

DATASRC=#ID

FRAME=ABOVE | BELOW | BORDER | BOX | INSIDES | LHS | RHS | VOLD | VSIDES

HEIGHT=n

ID=value

LANG=language

LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS

STYLE=cssl-properties

TITLE=text

WIDTH=n

event = script>

Наведемо опис атрибутів, що найчастіше використовуються.

- ALIGN — задає вирівнювання таблиці. За замовчуванням — по лівій межі.
- BACKGROUND — URL фонового зображення.

 BGCOLOR, BORDERCOLOR, BORDERCOLORDARK, BORDERCOLORLIGHT — колір фону, рамки, тіні рамки і її підсвітки (останні два атрибути використовуються разом з атрибутом BORDER).

 BORDER — товщина рамки у пікселях.

 CELLPADDING — відстань у пікселях між даними у клітинках і її межами.

 CELLSPACING — відстань у пікселях по вертикалі і горизонталі між клітинками.

 COLS — кількість стовпчиків у таблиці.

 FRAME — задає ті частини рамки таблиці, які будуть відображатися на екрані. Допустимі значення:

- BORDER (відображається уся рамка, використовується за замовчуванням);
- VOLD (рамка без зовнішніх меж);
- ABOVE (нема рамки в основі таблиці);
- BELOW (нема рамки у верхній частині таблиці);
- HSIDES (нема рамки на межах);
- LHS (нема рамки на правій межі );
- RHS (нема рамки на лівій межі);
- VSIDES (нема рамки в основі та у верхній частині таблиці);
- BOX (відображається уся рамка).

 HEIGHT і WIDTH — висота і довжина таблиці. Встановлюється в пік селях або у відсотках.

 RULES —задає ті внутрішні роздільні лінії, які будуть відображатися.

Допустимі значення:

- NONE (лінії не відображаються);
- GROUPS (відображаються горизонтальні лінії між групами елементів у таблиці);
- ROWS (відображаються горизонтальні лінії);

- COLS (відображаються вертикальні лінії);
- ALL (відображаються усі лінії).

Синтаксис тега <CAPTION>:

```
< CAPTION
      ALIGN=CENTER | LEFT | RIGHT | TOP
      CLASS=classname
      ID=value
      LANG=language
      LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
      STYLE=cssl-properties
      TITLE=text
   VALIGN=BOTTOM | TOP
      event=script>
```

Таблицю можна створити командою **Table | Insert |Table**. Розглянемо приклад створення телефонів й адрес електронної пошти.

```
<HTML>
<HEAD><TITLE>Телефони та адреси електронної пошти </ TITLE ></ HEAD>
<BODY>
<TABLE border=1 width=”75%”>
<CAPTION valign=”top”>
  Телефони та електронна пошта
</CAPTION>
  <CAPTION valign=”top” Телефони та електронна пошта </CAPTION>
   <TR>
        <TD></TD>
        <TH>Телефони</TH >
        <TH>e-mail</TH >
  </TR>
  <TR>
        <TD>Сундуков</TD>
        <TD>235-28-46</TD>
```

```
        <TD><A href=”mailto:syndykov@factor.ua”> syndykov@factor.ua </A></TD>
  </TR>
  <TR>
        <TD>Сідоренко</TD>
        <TD>456-15-54</TD>
        <TD><A href=”mailto:IIS@IS4556.spb.edu”> IIS@IS4556.spb.edu </A></TD>
  </TR>
  <TR>
        <TD>Кличко </TD>
        <TD>345-79-46</TD>
        <TD><A href=”mailto:KLF@pepci.ru”></A> KLF@pepci.ru </TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

Розглянемо створення списку з користувальним маркером.

```
<HTML>
<HEAD><TITLE> список з користувальним маркером</ TITLE ></ HEAD>
<BODY>
<TABLE border=0 cellPadding=1 cellSpacing=1 width=”75%”>
   <TR>
        <TD><IMG src=”Text.gif” alt=&#149;></TD>
        <TD>Створення таблиці в HTML</TD>
  </TR>
  <TR>
        <TD>< IMG src=”Text.gif” alt=&#149;> </TD>
  </TR>
```

```
</TABLE>
</BODY>
</HTML>
```

Розглянемо вирівнювання елементів у формі за допомогою таблиці

```
<HTML>
<HEAD><TITLE> Бланк замовлення</ TITLE ></ HEAD>
<BODY>
<FONT size=5><STRONG> Бланк замовлення</ STRONG >
</FONT>
<P></P>
<TABLE border=0>
  <TR>
      <TD>ПІБ</TD>
      <TD colspan=3><INPUT name=”text1” size=40 style=”height: 22px; width: 289px”>
</TD>
      <TD></TD>
  </TR>
  <TR>
      <TD>Адрес</TD>
      <TD colspan=3><INPUT name=”text2” size=40 style=”height: 22px; width:
289px”></TD>
      <TD></TD>
      <TD></TD>
 </TR>
  <TR>
      <TD>Місто</TD>
      <TD><INPUT name=”text3” size=22></TD>
```

```
        <TD>Код</TD>
        <TD><INPUT name=”text4” size=8 style=”height: 22px; width: 79px”></TD>
  </TR>
</TABLE>
</BODY>
</HTML>
```

## 12.6. Фрейми

Фрейми дозволяють поділяти сторінку на різні області перегляду, кожна з яких є HTML-документ. Визначення змісту фрейма починається з парного тега <FRAMESET>. В його області дії за допомогою тега < FRAME> задаються інші фрейми. Важливим атрибутом тега < FRAME> є **scr,** який задає URL документа, що розміщується у фреймі. Синтаксис тега <FRAMESET> і <FRAME>:

```
<FRAMESET
BORDER=pixels
BORDERCOLOR=color
CLASS=classname
COLS=col-widths
FRAMEBORDER=NO | YES | 0 | 1
FRAMESPACING=spacing
ID=value
LANG=language
LANGUAGE=JAVASCRIPT |JSCRIPT | VBSCRIPT | VBS
ROWS=row-heights
TITLE=text>
```

- COLS і ROWS — визначають кількість частин, на які ділиться вікно браузера по горизонталі та вертикалі, а також їх розміри.
- BORDER, BORDERCOLOR, FRAMEBORDER — задають товщину, колір рамки фрейма, а також її видимість.
- FRAMESPACING — відстань між кадрами у пікселях.

Синтаксис тега < FRAME>:

<FRAME

BORDERCOLOR=color

CLASS=classname

DATAFLD=colname

DATASRC=#ID

FRAMEBORDER= NO | YES | 0 | 1

HEIGHT=n

ID=value

LANG=language

LANGUAGE=JAVASCRIPT |JSCRIPT | VBSCRIPT | VBS

MARGINHEIGHT=pixels

MARGINWIDTH=pixels

NAME=window_name | _blank | _parent | _self | _top

NORESIZE=NORESIZE | RESIZE

SCROLLING=AUTO | NO | YES

SCR=url

TITLE=text

WIDTH=n

event=script>

- MARGINHEIGHT, MARGINWIDTH — довжина в пікселях внутрішньої межі фрейма.
- NORESIZE — користувач не може змінити розмір фрейма.
- SCROLLING — керує зображенням смуги прокрутки.

Розглянемо створення у вікні броузера трьох фреймів. Ліву частину екрана займає документ, що складає 1/5 частину вікна, а права частина екрана, що складає 4/5 вікна броузера, розділена по вертикалі на дві рівні частини. Кожній із трьох частин вікна броузера відповідає свій HTML-документ (FrameLeftSide.htm, FrameUp.htm і FrameDown.htm), в якому відображаються відповідні роз'яснювальні написи.

<HTML>

<HEAD><TITLE>Фрейми </ TITLE ></ HEAD>

<FRAMESET cols="*,4*">

```
<FRAME frameborder=”yes” src=”FrameLeftSide.htm”>
<FRAMESET row=”50%, 50%”>
< FRAME frameborder=”yes” src=FrameUp.htm>
<FRAME frameborder=”yes” src=FrameDown.htm>
</FRAMESET >
</FRAMESET >
</HTML >
```

*Організація гіперпосилань у фреймах*

Для організації навігації між фреймами треба присвоїти їм унікальні імена. Це робиться встановленням значення атрибута name тега <FRAME>. При цьому в тезі <A> створення гіперпосилання треба використовувати не тільки атрибут href для задання посилання на документ, але і атрибут target для вказування імені фрейма, в який буде завантажуватися документ.

Розглянемо створення гіперпосилання у фреймах. Вікно броузера складається із трьох фреймів: лівого, правого верхнього і правого нижнього. При завантаженні документа FrameLeftSide1.htm в лівому фреймі виводиться посилання на два документа. При натисненні мишею на правому верхньому із них завантажується документ FrameUp.htm, а при натисненні мишею на другому з них — документ FrameDown.htm.

Створимо код документа, що створює фрейми.

```
<HTML>
<HEAD><TITLE>Організація гіперпосилань</ TITLE ></ HEAD>
<FRAMESET cols=”2*, 3*”>
<FRAME src=”FrameLeftSide1.htm”>
<FRAMESET row=”50%, 50%”>
< FRAME name=“fraUP” frameborder=”yes”>
<FRAME name=“fraDown” frameborder=”yes”>
```

```
    </FRAMESET >
</FRAMESET >
</HTML>
```

Створимо код документа FrameLeftSide1.htm, що завантажений у лівий фрейм.

```
<HTML>
<HEAD><TITLE>Лівий фрейм </ TITLE ></ HEAD>
<BODY>
Завантажити документ в
<UL>
    <LI><A href=”FrameUp.htm” target=”fraUP”>верхній</A>фрейм
    <LI><A href=”FrameDown.htm” target=”fraDown”>нижній </A>фрейм
</UL>
</BODY>
</HTML>
```

## 12.7. Каскадна таблиця стилів

Стилі за способом використання розділяються на три групи:

- Задавання стилів для окремого фрагмента документу (Inline Style).
- Вбудовування таблиці стилів (Embedding) в документ за допомогою парного тега <STYLE>.
- Зв’язування (Linking Style) документа з таблицею стилів, яка зберігається в окремому файлі. Файл із стилями, як і HTML-файл, є текстовим файлом з розширенням .css. Це дозволяє виконувати управління CSS-стилями на всіх Web-сторінках проекту.

*Задавання стилів для окремого фрагмента документу* (Inline Style) виконується встановленням значення атрибута STYLE тега цього фрагменту. Синтаксис:

<tag STYLE =”attribute:value; attribute:value; …”></tag>

В залежності від конкретної ситуації для встановлення стилю слід застосувати один із наступних підходів.

- Зміна стилю всередині тега виконується встановленням значення атрибута STYLE цього тега. Наступна інструкція зображає заголовок H1 курсивним шрифтом темно-синього кольору.

<H1 style=”font- style: italic; color: navy”>Новий стиль заголовку</H1>

- Парний тег <DIV> дозволяє виділити фрагмент документа як один об‘єкт. Використання атрибута style тега <DIV> дозволяє управляти стилем цього фрагмента. Наступний приклад демонструє, як для заголовка і параграфа встановлюється загальний стиль: вирівнювання по центру, а шрифт — підкреслений.

```
<DIV style=”text-align: center; text-decoration: underline>”
<H1>Демонстрація</H1>
<P>
Використання тега &lt;DIV&gt.
</P>
</DIV>
```

- Парний тег <SPAN> дозволяє виділити невеликі фрагменти документу. За допомогою цього тега можна виділити окреме слово або групу слів абзацу. Використання атрибута STYLE тега <SPAN> дозволяє управляти стилем цього фрагмента. В наступному прикладі одне слово виділяється жовтим кольором.

```
<P> <SPAN style=”color: yellow”>жовтим</SPAN>кольором</P>
```

*Вбудована таблиця стилів* задається в блоці стилів, який розташовується між парними тегами <STYLE> і розміщується в HEAD–блоці документу. Блок стилів складається із інструкцій, які визначають стилі для окремих елементів або груп елементів документу. Інструкції, що задають стиль, мають такий синтаксис:

*selector {attribute:value; attribute:value; …}*

де *selector* — селектор, який ідентифікує елемент або групу елементів документа, *attribute* і *value* задають атрибути стилю та їх значення.

Тег <STYLE> має такий синтаксис:

<STYLE
DISABLED
MEDIA=SCREEN | PRINT | ALL
TITLE=text
TYPE=”text/css”>

Обов’язковим атрибутом тега <STYLE> є TYPE. Цей атрибут задає тип MIME (Multipurpose Internet Mail Extension, стандарт електронної пошти Internet). Атрибуту треба присвоїти значення text/css для того, щоб броузери, які не підтримують вбудовані таблиці стилів, змогли їх ігнорувати.

Розглянемо як створити вбудований стиль в документ.

```
<HTML>
<HEAD>
<STYLE type=”text/css”>
<!- -
        BODY (backgroud: yellow; color; green)
        /* Жовтий фон документа з зеленими літерами */
        H1 {font: 20pt Arial bold; color: red}
        /* Заголовок H1: напівжирний шрифт Arial висотою 20 червоного кольору */
        P { font: 12pt Arial; text-indent: 1.25cm}
        /* Параграф: червоний рядок в 1.25 см, шрифт Arial висотою 12 */
- - >
</STYLE >
</HTML>
<BODY >
<H1>Новий стиль заголовка/H1>
<P>
        Новий стиль параграфа. Новий стиль параграфа. Новий стиль параграфа.
        Новий стиль параграфа. Новий стиль параграфа. Новий стиль параграфа.
```

</P>

</BODY>

</HTML>

*Примітка:*

Коментарі всередині таблиці стилів розташовуються між символами /* та */ і ігноруються броузером.

*CSS і MS FrontPage*

Програма MS FrontPage пропонує розробнику нові засоби роботи. Виберіть команду Format | Style. З'явиться діалогове вікно Style. В вікні Style у списках Styles відображаються усі доступні теги HTML. Кожний із цих тегів може бути використаний як назва селектора. Щоб відобразити вже створені вами правила CSS для Web-сторінки, в списку List виберіть User-defined styles. При виборі будь-якого із стилів списку Styles в області Paragraph preview приводиться зразок параграфа. В області Character preview демонструється, як буде виглядати вибраний символ стилю, . В області Description виводиться текстовий опис атрибутів стилю. Вилучати можна тільки стилі, що створені користувачем, натискуючи кнопку [Delete]. Для створення нового стилю треба натиснути кнопку New, і в результаті на екрані відобразиться вікно New Style. Для зміни вибраного стилю — кнопку Modify, і в результаті на екрані відобразиться вікно Modify Style, яке має ту ж структуру, що і вікно New Style. Натиснути кнопку Format. З'явиться список з 5-ма елементами: Font, Paragraph, Border, Numbering і Position, кожний із яких у діалоговому режимі дозволяє встановити необхідні правила модифікованого стилю.

Для встановлення зв'язку таблиці стилів з документом треба застосувати тег <LINK>, який повинен розташовуватися між парними тегами <HEAD>. Таблиця стилів зберігається в текстовому css-файлі.

Коли треба визначити один і той же стиль для кількох селекторів, то можна об’єднати їх в інструкції визначеного стилю в групу селекторів. При цьому селектори розділяються комою. Наприклад:

P, L1 {font-size: 12pt}

В інструкції визначення стилю допустимо групування атрибутів властивостей за їх типом (border, background, font, list, margin, padding). Наприклад, наступні дві інструкції задають один і той же стиль.

P{font-size: 12pt; font-weight: bold; font-family: courier}

P{font: 12pt bold courier}

На практиці виникають випадки, коли один стиль застосовується у контексті іншого. Наприклад, заголовок H2 повинен мати темно-синій колір тільки у тому випадку, коли його шрифт курсивний. В таких ситуаціях треба використовувати контекстний селектор. В контекстному селекторі перераховуються селектори окремих стилів, які входять у контекст, а роздільником між ними використовується пропуск. Продемонструємо як визначається контекстний стиль на прикладі, який був описаний.

```
< STYLE >
<! - -
        H2 I {color: navy}
- >
</STYLE >
<BODY>
<H2>Це чорний колір.</H2>
<H2><I>Це темно-синій колір.</I ></H2>
</BODY>
```

Додаткову гнучкість для застосування таблиці стилів дає можливість визначення в ній різних класів стилів для одного й того ж елемента, які потім можна застосувати до елементів документу.

□ Border — заголовок обрамляється червоною рамкою і вирівнюється по центру.

□ Background — заголовок вирівнюється по центру і має світло-зелений фон.

Розглянемо в наступному коді два класи заголовка H1:

```
<HTML>
<HEAD>
<STYLE type=”text/css”>
<!- -
        H1.Border {border-width: border: solid; border-color: red; text-align: center}
        H1.BackGround {background-color: lime; text-align: center}
- - >
</STYLE>
</HEAD>
<BODY>
        <H1 class=”Border”> Заголовок H1 з рамкою</H1>
        <H1 class=” BackGround”> Заголовок H1 з фоном</H1>
        <H1> Стандартний заголовок H1 </H1>
</BODY>
</HTML>
```

HTML дозволяє розташувати будь-який елемент у тому місці документу, де побажає розробник. Для цього треба застосувати наступні властивості атрибута STYLE:

□ position — задає абсолютне або відносне місцезнаходження елемента в документі. Допустимі значення:

absolute, тоді властивості left і top задають в пікселях абсолютне розміщення елемента в документі;

relative, тоді властивості left і top задають в пікселях відносне розміщення елемента в документі;

static — елемент розміщується в документі так, як при відсутності значення властивості position .

□ left, top, width, height — координати верхнього лівого кута елемента, його довжина і висота.

□ z – index — розміщує елемент спереду або позаду всіх елементів, що з ним перетинаються. Допустимі значення: auto або ціле число, де 0 відповідає розміщенню елемента на задньому плані.

Використання атрибута ID в якості селектора стилю.

При визначені стилю id, який застосовується тільки до одного елемента, спочатку треба розташувати знак #.

Наприклад:

```
<STYLE>
    #idP {text-indent: 10px; font-size: 12pt}
</STYLE>
<BODY>
    <P id=”idP”>
    це <SPAN id=”idBI”>слово</SPAN>набрано напівжирним курсивом</P>
</BODY
```

*Фільтри*

Фільтри дозволяють перетворювати усі теги, у яких атрибут STYLE має властивість filter, наприклад, <BODY>, <BUTTON>, <DIV>, <IMG>, <INPUT>, <MARQUEE>, <SPAN>, <TABLE>, <TD>, <TEXTAREA>, <TFOOT>, <TH.>, <THEAD>, <TR>. При цьому у тегів <DIV> і <SPAN> треба вказувати точно місцезнаходження або їх розміри. Синтаксис:

{filter: filtername (fparameter1, fparameter2…)}

де filtername — ім’я фільтра, а fparameter — його параметри.

Перерахуємо основні фільтри

## 12.8. Елемент керування WebBrowser

Елемент керування **WebBrowser** дозволяє додавати можливості броузера. Наприклад, його можна використовувати для відображення ресурсів локального комп'ютера.

| **Фільтр** | **Опис** |
|---|---|
| Alpha | Встановлює рівень прозорості елемента |
| Blur | Розмитий образ, що створює ефект руху з великою швидкістю |
| Chroma | Робить вказаний колір прозорим |
| DropShadow | Створює тінь у вигляді офсетного образу зображення |
| FlipH, FlipV | Створює горизонтальний і вертикальний дзеркальний образ |
| Glow | Створює ореол навколо об'єкта |
| Grayscale | Переводить відображення в чорно-білий формат |
| Invert | Обертає компоненти і інтенсивність кольору |
| Wave | Створює хвилеподібне перекручення образу вздовж осі абсцис |
| Xray | Створює чорно-білий рентгенівський образ елемента |

Для додавання елемента керування **WebBrowser** у панель інструментів, треба виконати команду **Tools | Additional Controls**. У вікні **Дополнительные элементы управления** у списку **Доступные элементы управления** введіть прапорець **Обозреватель веб-страниц Micrisift** і натисніть кнопку **ОК**.

DLL-файлом елемента керування **WebBrowser** є Shdocvw.dll. Елемент керування **WebBrowser** створюється кнопкою **WebBrowser.**

Перерахуємо основні властивості елемента керування **WebBrowser**.

| **Властивості** | **Опис** |
|---|---|
| Busy | Вказує, чим зайнятий елемент керування: завантаженням із мережі чи виконує інші операції |
| LocationName | Повертає рядок, який містить ім'я поточного ресурсу, що |

| | |
|---|---|
| | відображається в елементі керування |
| LocationURL | Повертає рядок, який містить URL поточного ресурсу, що відображається в елементі керування |

Перерахуємо основні методи елемента керування **WebBrowser**.

| **Метод** | **Опис** |
|---|---|
| GoBack | Переходить до попереднього елемента із списку історії |
| GoForward | Переходить до наступного елемента із списку історії |
| GoHome | Переходить до домашньої сторінки |
| GoSearch | Переходить до поточної сторінки пошуку |
| Navigate | Переходить до вказаного ресурсу, що заданий за допомогою URL або повного шляху |
| Refresh | Перезавантажує поточний ресурс, що відображається в елементі керування |
| Refresh2 | Перезавантажує поточний ресурс, що відображається в елементі керування. Синтаксис:<br>Refresh2 [*Level*]<br>де *Level* задає рівень оновлення. Допустимі значення:<br>REFRESH_NORMAL або 0 – швидка регенерація без відсилання на сервер HTTP-заголовка “pragma:nocache”;<br>REFRESH_IFEXPIRED або 1 – швидка регенерація;<br>REFRESH_COMPLETELY або 3 – повна регенерація |
| Stop | Зупиняє поточне переміщення або завантаження |
| ExecWB | Підтримує такі команди роботи з файлами, як Print, Print Preview, Save, Save As, New з відображенням відповідних вікон.<br>Синтаксис:<br>ExecWB *nCmdID, nCmdExecOpt,* [*praIn*], [*praOut*]<br>*nCmdID* – ідентифікатор команди. Наприклад,<br>◦ OLECMDID_OPEN або 1, OLECMDID_SAVE 3,<br>◦ OLECMDID_SAVEAS або 4,<br>◦ OLECMDID_PRINT або 6. |

| | |
|---|---|
| | *nCmdExecOpt* – режим виконання команди. Допустимі значення:<br>◦ LECMDEXECOPT_DODEFAULT або 0,<br>◦ LECMDEXECOPT_PROMPTUSER або 1,<br>◦ LECMDEXECOPT_DONTPROMPTUSER або 2,<br>◦ LECMDEXECOPT_SHOWHELP або 3.<br>*praIn, praOut* – вхідний і вихідний параметр команди. |

Приклад процедури, що здійснює друк документу, який завантажено в елемент керування **WebBrowser:**

```
Private Sub cmdPrint_Click ()
On Error GoTo ErrorHandler:
WebBrowser1.ExecWB OLECMDID_PRINT, _
    LECMDEXECOPT_ PROMPTUSER, “”, “”
ErrorHandler:
End Sub
```

Перерахуємо основні події елемента керування **WebBrowser.**

| **Подія** | **Опис** |
|---|---|
| BeforeNavigate2 | Генерується до того, як **WebBrowser** переміститься до іншого URL. |
| NavigateComplete2 | Генерується по завершенню переміщення **WebBrowser** на новий URL |
| DownloadBegin, ProgressChange DownloadComplete | Генерується при початку завантаження документа в **WebBrowser,** у процесі завантаження та по її завершеню. Процедура ProgressChange має два параметри: Progress (кількість вже завантажених даних), ProgressMax (загальна кількість даних, що буде завантажено). |

Наступний код демонструє, як обробка події BeforeNavigate2 дозволяє створити обмеження на ті документи, до якх можна переміщатися (у даному випадку вони повинні бути HTML-документами).

```
Private Sub WebBrowser1_BeforeNavigate2 ( _
        ByVal pDisp As Object, URL As Variant, _
        Flags As Variant, TargetErameName As Variant, _
        PostData As Variant, Headers As Variant, _
        Cancel As Boolean)
        If Left (Lcase (URL), 3) <> “htm” Then
                Cancel = True
                MsgBox “Access denied to URL: ” & URL
        End If
End Sub
```

Наступні дві процедури демонструють події DownloadBegin і ProgressChange.

```
Private Sub WebBrowser1_DownloadBegin ()
        Label1.Caption = “Begin Downloading”
End Sub
Private Sub WebBrowser1_ProgressChange _
        (ByVal Progress As Long, _
        ByVal ProgressMax As Long)
        If ProgressMax > 0 Then
                Label1.Caption = Progress / ProgressMax
        Else
                Label1.Caption = “Ready”
        End If
End Sub
```

## 12.9. Міні-броузер

Як перший Web-додаток  створимо проект, який дозволяє переміщуватися по Web-документах, що зберігаються в мережі або на вашому комп’ютері.

Створимо форму, на якій розташуємо поле вводу і елемент керування WebBrowser. В модулі форми треба набрати код, що забезпечить відображення Web- документа у вікні елемента керування WebBrowser, URL-адреса якого вказана в полі вводу. Для тестування проекту, що створений в полі вводу введемо, наприклад, адресу http://www.microsoft.com/rus/.

Процедура першого вашого міні-броузера:

```
Private Sub TextBox1_Change ()
    If Len (TextBox1.Text) > 0 Then
        WebBrowser1.Navigate TextBox1.Text
    End If
End Sub
```

## 12.10. Питання для самоконтролю

1. Яке призначення мови гіпертекстової розмітки HTML?
2. Які основні важливі характеристики мови розмітки?
3. Що таке гіпертекст?
4. Що таке гіперпосилання?
5. Що таке дескриптор (тег)?
6. Що таке мова HTML?
7. Що таке HTML-файл?
8. Що таке броузер (browser)?
9. В якому додатку можна скласти код HTML-документа?
10. Яка використовується програма для створення інтерактивної Web-сторінки?
11. Яку програму треба запустити для створення HTML-документа?
12. З яких закладок складається вікно проекту?
13. Що можна застосовувати в закладці Normal вікна проекту?
14. Що можна робити розробнику Web-сторінки в закладці HTML вікна проекту?

15. Що можна робити розробнику в закладці Preview File/in Browser вікна проекту?

16. Що таке адреса URL (Uniform Resource Locator)?

17. З яких чотирьох частин складається адреса URL?

18. Що таке протокол в адресі URL?

19. Що розташовано на ИмяСервера в адресі URL?

20. Чому дорівнює НомерПорта в адресі URL?

21. На які три частини умовно можна розбити теги HTML?

22. Що таке парний тег?

23. Що йде після заголовка в HTML-документі?

24. Яким тегом повинно починатися і закінчуватися тіло в HTML-документі?

25. Які теги використовуються для заголовку HTML-документа?

26. Які теги використовуються для тіла HTML-документа?

27. Теги, що використовуються при форматуванні HTML-документа.

28. Тег зображення в HTML-документі.

29. Тег вставки гіперпосилання і закладки.

30. Списки.

31. Таблиці

33. Фрейми

34. Організація гіперпосилань у фреймах

35. Каскадна таблиця стилів

36. MS FrontPage.

37. *Елемент керування WebBrowser*

38. *Міні-броузер*

## 13. СТВОРЕННЯ ІНТЕРАКТИВНИХ WEB-СТОРІНОК

На основі мови сценаріїв VBScript створимо динамічні й інтерактивні Web-сторінки. VBScript — це мова розробки Web-сценаріїв. За допомогою VBA можна сконструювати повнофункціональні офісні проекти, але цю мову не

можна використовувати в Web-документах. В цьому випадку на допомогу приходить VBScript, який дозволяє:

□ впроваджувати в HTML-документ програми обробки з боку клієнта.

□ отримувати доступ до властивостей, методів і подій:

- елементів HTML-документа;
- ActiveX-елементів HTML-документа;
- об’єктів Internet Explorer.

## 13. 1. Середовище розробки VBScript-сценаріїв

Броузер, що має VBScript як інтерпретатор (наприклад, Microsoft Internet Explorer), може запускати програми, що написані на VBScript.

Мова Microsoft FrontPage пропонує розробнику VBScript-сценаріїв зручне середовище для написання коду — Microsoft Development Environment. Для виклику Microsoft Development Environment в Microsoft FrontPage достатньо виконати команду **Tools | Macro | Microsoft Script Editor** [Alt+Shift+F11] (рис. 3.2.).

Код VBScript розташовується між парними тегами <SCRIPT>. Атрибут LANGUAGE цього тега вказує, якою мовою написано сценарій (в нашому випадку це VBScript).

В загальному випадку тег <SCRIPT> має таиий синтаксис:

```
<SCRIPT
    CLASS=classname
    DEER
    EVENT=eventname
    FOR=element
    ID=value
    LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
    SKC=url
    TITLE=text
    TYPE=MIME_type
```

>

Крім того, код сценаріїв встановлюється в теги коментарів тому, що не всі броузери його впізнають.

*Примітка.* В VBScript є єдиний тип даних — Variant, змінні мають дві області видимості: процедуру і модуль, змінні об'являються тільки операторами Dim і Public без вказівок на тип даних.

## 13. 2. Об'єктна модель Internet Explorer

При напису сценаріїв використовується об'єктна модель Internet Explorer. Перерахуємо основні об'єкти цієї моделі (табл.. 11.1).

| **Об'єкт** | **Опис** |
|---|---|
| window | Об'єкт верхнього рівня – Internet Explorer |
| frame | Фрейм. всі фрейми створюють сімейство Fremes |
| history | Призначений для навігації по списку, що розглянуті у даному сеансі сторінок |
| navigator | Призначений для збереження інформації про броузер |
| location | URL поточної сторінки |
| screen | Видає інформацію про екран поточного користувача |
| script | Зберігає код сценаріїв даного вікна |
| event | Видає інформацію, яка пов'язана з подіями в сценарії |
| parent | Вихідний об'єкт window |
| document | Область, в якій сценарій виводить інформацію |
| link | Вкладається в об'єкт document. Призначений для збереження в документі посилань |
| anchor | Вкладається в об'єкт document. Призначений для збереження в документі посилань, які задані тегом <A> |
| form | Вкладається в об'єкт document. Подається як форма |
| element | Вкладається в об'єкт form. Подається як об'єкт форми |

### 13. 2. 1. Об’єкт window

При написанні коду об’єкт window посилання на об’єкт window можна пропустити (за замовчуванням).

Перерахуємо події, що підтримуються об’єктом window.

| **Події** | **Момент генерації** |
|---|---|
| onbeforeunload | Перед закриттям вікна броузера |
| jnblur | При втраті вікном фокусу |
| onerror | При помилці |
| onfocus | При отримані вікном фокусу |
| onhelp | При натисненні клавіші [F1] |
| onload | При відкриванніі вікна броузера |
| onresize | При зміні розмірів вікна |
| onscroll | При прокручуванні документа |
| onunload | При закриванніі вікна броузера |

Перерахуємо основні властивості об’єкта window.

| Властивість | Опис |
|---|---|
| defaultStatus | Повідомлення, що відображається за замовчуванням у рядку стану вікна броузера |
| status | Повідомлення, що відображається у рядку стану вікна броузера |
| document, event, navigator, screen | Повертаємо об’єкти document, event, navigator, screen |

Порахуємо основні методи об’єкта window.

| **Метод** | **Опис** |
|---|---|
| open | Відкриває нове вікно броузера |
| close | Закриває вікно броузера |
| execScript | Виконує вказаний код |
| showModalDialog | Відкриває модальне вікно для перегляду HTML-документу |
| alert | Відображає діалогове вікно, яке аналогічне вікну MsgBox з повідомленням і кнопкою ОК |
| confirm | Відображає діалогове вікно, яке аналогічне вікну MsgBox з повідомленням і кнопкою ОК і Cancel. При натисненні кнопки ОК повертає значення True, а кнопки Cancel – False |
| prompt | Відображає діалогове вікно введення, яке аналогічне вікну InputBox, і повертає введенні значення. |
| setInterval | Створює таймер, який виконує код через вказаний часовий інтервал |
| clearInterval | Припиняє роботу таймера, який створений методом setInterval, і знищує його із пам’яті. |
| setTimeout | Створює таймер, який виконує код тільки один раз після закінчення інтервалу часу |
| clearTimeout | Припиняє роботу таймера, який створений методом setTimeout, і знищує його із пам’яті. |
| scrollBy, scrollTo | Прокручувати документ на вказану відстань і до вказаного місця розташування |

*Біжуче повідомлення в рядку стану*

Як приклад використання властивості status разом з методом setInterval об’єкта window створимо біжуче повідомлення в рядку стану. Даний ефект створюється шляхом додавання на початку рядка ще одного рядка, який складається із пропусків з наступним зменшенням кількості цих пропусків.

Програмний код має такий вигляд:

```
<HTML>
<SCRIPT language = ”VBScript”>
<!—
      Public I, strRun, Num
      Const strInit = “”<<Andrey Garnaev
      Sub Window_OnLoad ()
            window. setInterval “RunStatus (), 100
strRun = space (n) & strInit
Num = Len(strInit) + n
I = Num
      End Sub
      Sub RunStatus ()
            If I > n Then
                  window.status = space (I) & Left(strInit, Num  - I)
                  I = I – 1
            ElseIf  0 < I Then
                  window.status = space (I) & strInit
                  I = I – 1
            Else
                  I = Num
            End If
      End Sub
-->
</ SCRIPT>
<BODY>
      <H!> Біжуче повідомлення в рядку стану</H1>
</BODY>
</HTML>
```

*Об’ява змінних і констант рівня модуля*

Змінні рівня модуля треба  об’являти перед скриптами, які повинні отримувати до них доступ. Тому <HEAD>-блок є найліпшим місцем для їх розміщення.

### 13. 2. 2. Об’єкт document

Об’єкт document — це документ, що міститься у вікні броузера.

Перерахуємо основні події об’єкта document.

| **Події** | **Момент генерації** |
|---|---|
| onafterupdate | Після оновлення документу |
| onbeforeupdate | Перед оновленням документу |
| onclick | При клацанні кнопкою миші |
| ondblclick | При подвійному клацанні кнопкою миші |
| ondragstart | На початку операції буксируванні |
| onhelp | При натисненні клавіші <F1> |
| onkeydown | При натисненні клавіші |
| onkeypress | При натисненні клавіші, але після подій onkeydown і onkeyup |
| onkeyup | При відпусканні клавіші |
| onmousedown | При натисненні кнопки миші |
| onmousemove | При переміщені покажчика миші |
| onmouseout | При виведені покажчика миші за межі документа |
| onmouseover | При виведені покажчика миші за межі документа, коли він переміщується |
| onmouseup | При відпусканні кнопки миші |
| onselectstart | На початку операції виділення |

Перерахуємо основні властивості об’єкта document.

| **Властивість** | **Опис** |
|---|---|
| bgColor,fgColor | Колір фону і переднього плану документа |
| title | Назва документа, який відображається в заголовку броузера |
| linkColor, | Колір гіперпосилань, активних і які вже зустрічалися на |

| | |
|---|---|
| alinkColor, vlinkColor | протязі даного сеансу роботи з документом |
| fileCreateDate | Дата створення документа |
| fileModifieDate | Дата останньої модифікації документа |
| fileUpdatedDate | Дата останньої модифікації документа сервером |
| fileSize | Розмір файлу |
| activeElement | Активний елемент документа |
| elementFromPoint | Повертає мій елемент документа, який знаходиться у точці з координатами x, y |
| all | Сімейство всіх елементів документа |
| anchors, forms, images, links, styleSheets | Сімейство закладок, форм, малюнків, посилань й таблиць стилів документа |
| body | Тіло документа |

Перерахуємо основні методи об’єкта document.

| **Метод** | **Опис** |
|---|---|
| open | Відкриває документ і дозволяє потоковий вивід |
| write | Записує рядок у документ |
| writeln | Записує рядок у документ і переводить покажчик на початок нового рядка. Переміщення покажчика здійснюється тільки в межах тексту, що розміщений між тегами <PRE> |
| close | Закриває потік даних, що спрямовані у документ, і оновлює екран |
| clear | Закриває потік даних, що спрямовані у документ, і очищає екран |

*Сімейство елементів*

Число елементів в сімействі повертається властивістю length.

Конкретний елемент сімейства повертається методом item або посиланням на елемент. Посилання на елемент сімейства можна здійснювати або по його номеру, або використовуючи значення атрибутів id і name, або безпосередньо по його імені. Наприклад, коли в документі у тега з номером 7 встановлено значення атрибута id рівним myHead, то приведені нижче 5 способів вказують на один й той же елемент.

- document.all (7)
- document.all.item (7)
- document.all (“myHead”)
- document.all.item (“myHead”)
- document.all. myHead

Наступна інструкція повертає значення атрибута id цього елемента

```
alert document.all.item (7). id
```

Різниця між атрибутами id і name у тому, що кожний елемент в документі має унікальне значення атрибута id, у той час як декілька елементів в документі можуть мати одне й те ж значення атрибута name.

Елементи сімейства мають властивість tagName, яка повертає ім’я його тега.

Наприклад, запропонований нижче код повертає число елементів документа і імена всіх його тегів.

```
alert document.all.length
Dim obj
For Each obj in document.all
    alert obj.tagName
Next
```

Існує ще одне важливе сімейство tag. Наприклад, наступний код перевіряє, чи містяться в документі малюнки. Коли їх немає, то виводиться повідомлення.

```
If document.all.tags (“IMG”).length = 0 then
    alert “В документі нема малюнків”
End if
```

*Сторінка, яка динамічно змінюється*

Розглянемо приклад коду, який інформує про час завантаження документа в броузер і в залежності від цього часу відображає або повідомлення “Ще не вечір!”, або, коли вже пізно, повідомлення, що пора кінчати роботу — “Пора це кінчати!”. Після цього екран поступово темніє, поки не стане чорним. Потім на екрані з’являється повідомлення “На добраніч”.

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” for = “window” event = “OnLoad”>
! –
      Dim h
      h = Hour (Now)
      document.open
      document.write “<BODY bgcolor = yellow>”
      document.write “<H1> Час завантаження документу: ” & time () &
“</H1>”
      If h >= 4 Then
            document.write “<H1>  Ще не вечір! </H1>”
      Else
            document.write “<H1> Пора це кінчати! </H1>”
            Dim I
            For I = 0  to 255
                  document.bgColor = rgb (0, 255 – I, 255 - I)
            Next
                  document.close
                  document.write “<BODY bgcolor = black>”
                  document.write “<FONT color = white>”
                  document.write “<H1> На добраніч! </H1>”
            End If
            document.close
-->
</SCRIPT>
</HEAD>
```

```
</HTML>
```

*Примітка.* В VBScript в операторі циклу після ключового слова Next ім'я змінної в прикладі не вказано. В VBA в операторі циклу після ключового слова Next ім'я змінної можна вказувати, а можна і не вказувати.

*Прокручування документу*

Можна автоматично прокручувати документ. Для цього при завантаженні документа в броузер включається таймер, який кожні 0.1 сек прокручує документ вниз на 10 пиксилів. Процес прокручування продовжується до кінця документу. Для реалізації прокрутки документу використовуються такі властивості об'єкта body.

| **Властивість** | **Опис** |
|---|---|
| scrollTop, scrollLeft | Поточна координата верхнього лівого кута тієї частини документа, яка відображається у вікні броузера |
| clientHeight,clientWidth | Висота і довжина вікна броузера, в якому виводиться документ |
| scollHeight, scrollWidth | Висота і довжина документа |

Програмний код прокрутки документа

```
<HTML>
<HEAD>
<SCRIPT language = "VBScript" >
<! –
Public tmr, T, H
Sub window_onload ()
      tmr = window.setInterval ("Run", 100)
End Sub
Sub Run ()
      Dim sH
      window.scrollby 0, 10
      T = window.document.body.scrollTop
      H = window.document.body.elementHeight
```

```
        sH = window.document.body.scrollHeight
        window.status  = “Scrolling”
        If  T + H >= sH Then
              window.clearInterval (tmr)
              window.defaultStatus = “Ready”
        End If
End Sub
-- >
</SCRIPT>
</HEAD>
<BODY>
<FONT SIZE = 8>
<P>1</P><P>2</P><P>3</P><P>4</P><P>5</P>
<P>6</P><P>7</P><P>8</P><P>9</P><P>10</P>
<P>11</P><P>12</P><P>13</P><P>14</P><P>15</P>
<P>16</P><P>17</P><P>188</P><P>19</P><P>20</P>
</BODY>
</HTML>
```

*Рухоме зображення і об’єкт* style

Розглянемо приклад рухомого зображення картинки (колобок, файл Dot1.gif) у вікні броузера по закону більярдної кулі,  на фоні  іншої картинки (Облака.gif). Виконанню цього процеса допоможе об’єкт *style*  зі своїми властивостями: left, top, height, width, які встановлюють місце знаходження елемента на екрані і його розміри.

Програмний код зображення, яке рухається.

```
 <HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<! –
Public intTimeout
Public x, y, xStep, wDot,hDot, w, h
Sub Window_OnLoad ()
```

```
        wDot = imgDot.width
        hDot = imgDot.height
        Window_OnResize
        imgDot.Stytle.left = x          'Поточна x-координата
        imgDot.Style.top = y            'Поточна y-координата
        Window.SetTimeout "MovingDot", 10
        xStep = (11 * rnd () + 5)  * (2 * rnd () - 1)
' Швидкість руху по осі x
        yStep = (11 * rnd () + 5)  * (2 * rnd () - 1)
' Швидкість руху по осі y
End Sub
Sub Window_OnResize ()
        imgClouds.style.top = 0
        imgClouds.style.left = 0
        h = document.body.offsetHeight
        w = document.body.offsetWidth
        imgClouds.style.height = h
        imgClouds.style.width = w
        Randomize
        y = (h - hDot) * rnd ()
        x = (w  - wDot) * rnd ()
End Sub
Sub MovingDot ()
        x = x + xStep
        y = y + yStep
        If x > w  - wDot or x < 0 Then
              xStep = -xStep
        End If
        If y > h  - hDot or y <o Then
              yStep = -yStep
        End If
        imgDot.Style.Left = x
    imgDot.Style.Top = y
```

```
  Window.SetTimeout “MovingDot”, 10
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<IMG id = “imgClouds” src = “Облака.gif” style = “position: absolute”>
<IMG id = “imgDot” src = “dot1.gif”
      style = “position: absolute; width: 60px; height: 60px;
      filter: Chroma (Color = write)”>
</BODY>
</HTML>
```

*Примітка*: Для визначення розміру документу в коді використовуються властивості offsetHeight і offsetWidth об’єкта body. Властивості offsetTop, offsetLeft, offsetHeight, offsetWidth елемента документа повертають його координати і розміри в пікселях по відношенню до родового елемента.

*Закріплення зображення в кутках вікна*

Розглянемо приклад руху малюнків у випадку прокрутки документа або зміни розмірів вікна броузера. Малюнки переміщуються таким чином. що постійно знаходяться в кутках вікна.

Програмний код:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<! –
Sub SetImages ()
      UL.style.top = document.body.scrollTop
      UL.style.left = document.body.scrollLeft
      UL.style.top = document.body.clientWidth  + document.body.scrollLeft – UR.width
      LL.style.left = document.body.clientHeight + document.body.scrollTop – LL.height
```

```
      LL.style.left = document.body.scrollLeft
      LR.style.top = document.body.scrollHeight + document.body.scrollTop –
LR.height
      LR.style.left = document.body.clientWidth + document.body.scrollLeft –
LR.width
End Sub
-->
</SCRIPT>
</HEAD>
<BODY onload = “SetImages ()
      onresize = “SetImages ()
      onscroll = “SetImages ()
      style = “margin-left: 45px; margin-right: 45px;”>
<H1>1</H1><H1>2</H1><H1>3</H1><H1>4</H1><H1>5</H1>
<H1>6</H1><H1>7</H1><H1>8</H1><H1>9</H1><H1>10</H1>
<H1>11</H1><H1>12</H1><H1>13</H1><H1>14</H1><H1>15</H1>
<IMG id = “UL” src = “dot1.gif” style = “position: absolute;”
      width = “40” height = “40”>
<IMG id = “UR” src = “dot1.gif” style = “position: absolute; ”
      width = “40” height = “40”>
<IMG id = “LL” src = “dot1.gif” style = “position: absolute;”
      width = “40” height = “40”>
<IMG id = “LR” src = “dot1.gif” style = “position: absolute;”
      width = “40” height = “40”>
</BODY>
</HTML>
```

*Рядок-пружина*

Розглянемо приклад зміни відстані між літерами., що створює ефект пружини. При цьому застосовується властивість letterSpacing об’єкта style,яке і встановлює розміри між літерами в HTML-елементі.

Програмний код:

```
<HTML>
<HEAD>
```

```
<SCRIPT language = “VBScript” >
<! –
      PublicSps (5)
      Sps (0) = “Opx” : Sps (3) = “4px”
      Sps (1) = “1px” : Sps (4) = “6px”
      Sps (2) = “2px” : Sps (5) = “8px”
      Dim st
      st = 1
      Public tmr
      Sub window_OnLoad ()
            tmr =window.setInterval (“Spring (), 100)
      End Sub
      Sub window_OnUnload ()
            window.clearInterval tmr
      End Sub
      Sub Spring ()
            hTxt.style.letterSpacing =Sps (I)
      I = I + st
      If  I >= 5 Or I <= 0 Then st = -st
      End Sub
-->
</SCRIPT>
</HEAD>
<BODY bgcolor = “black”>
      <H1 id = “hTxt” style = “color: yellow; text-align: right”>
      Andrey Garnaev </H1>
</BODY>
</HTML>
```

*Ініціалізація змінних рівня модуля*

Ініціалізація змінних рівня модуля здійснюється у тому сценарію, де зроблена об’ява змінних.

*Ще раз про рядок, де біжуче повідомлення*

Об’єкт style дозволяє спростити процес формування рядка, де біжуче повідомлення, шляхом зміни координат HTML-елемента. Для цього можна використати властивості pixelleft, pixeltop, pixelhtight, pixelwindth, які встановлюють і повертають місце розташування елемента на екрані та його розміри в пикселях. При цьому координати могуть мати від’ємні значення, що спрощує написання коду, який створює тексти, що випливають із-за межі екрану.

Програмний код:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<! –
      Sub window_OnLoad ()
            window.setTimeout “RunHead (), 100
      End Sub
      Sub RunHead ()
            hMy.style.pixelleft = hMy.style.pixelleft + 10
            If hMy.style.pixelleft > 0 Then Exit Sub
            setTimeout “RunHead (), 100
      End Sub
-->
</SCRIPT>
</HEAD>
<BODY bgcolor = navy>
      <H1 id = “hMy” style = “color:red; position: relative; left: -300px”>
      Andrey Garnaev
       </H1>
</BODY>
</HTML>
```

### 13. 2. 3. Об’єкт navigator

Об’єкт navigator видає інформацію про броузер. Перерахуємо його основні властивості.

| **Властивість** | **Опис** |
|---|---|
| AppCodeName | Кодове ім‘я версії броузера |
| AppName | Назва броузера |
| AppVersion | Версія броузера |
| AppMinorVersion | Друга цифра у номері версії броузера |
| UserLanguage, systemLanguage | Мова користувача і системи за замовчуванням |
| Platform | Платформа |
| cpuClass | Тип CPU |
| userAgent | Ім‘я користувача, який інсталює броузер |

*Створення сторінки з інформацією про броузер*

Розглянемо роботу з об’єктом navigator на прикладі створення Web-сторінки, яка видає повідомлення користувачу про броузер: назву і версію.

Програмний код.

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<! –
      Sub window_OnLoad ()
      document.open
      document.bgColor = “yellow”
      document.fgColor = “navy”
      document.write “<H1>Інформація про броузер </H1>”
      document.write “<HR>”
      document.write “<UL>”
      document.write “<LI> <I>appName: </I> “ & window.navigator.appName
```

```
        document.write “<LI> <I>appVersion : </I>” &
window.navigator.appVersion
        document.write “<LI> <I>userLanguage </I>” &
window.navigator.userLanguage
        document.write “<LI> <I>Platform: </I>” & window.navigator. platform
        document.write “<LI> <I>cpuClass: </I>” & navigator.cpuClass
        document.write “</UL>
        document.close
        End Sub
-->
</SCRIPT>
</HEAD>
</HTML>
```

### 13. 2. 4. Об’єкт event

Об’єкт event видає інформацію, що пов’язана з подіями в сценарії. Цей об’єкт доступний тільки у випадку виникнення послідовності подій.

Перерахуємо основні властивості об’єкта event.

| **Властивість** | **Опис** |
|---|---|
| scrElement | Повертає елемент, який першим генерував подію. Для ідентифікації елемента застосовують властивість tagName, яка повертає ім‘я тега, як це здійснюється в наступному прикладі:<br><BODY language = “VBScript”<br>onClick = “alert window.event.srcelement.tagName”><br><H1> Демонстрація </H1><br><P> Визначення тега, де було дроблення клацання. </P><br></BODY> |
| type | Повертає ім’я генерованої події, але без префікса on. |
| clientX, clientY | Горизонтальна і вертикальна координати покажчика миші відносно клієнтській області вікна броузера і екрана. |
| screenX, | Наступний код відображає координати покажчика миші у |

| | |
|---|---|
| screenY | рядку стану при переміщені йоого над документом.<br><HTML><br><SCRIPT language = “VBScript” ><br><! –<br>Sub document_onmousemove ()<br>window.status = “x: ” & window.event.screenX & _<br>“y: ” & window.event.screenY<br>End Sub<br>Sub document_onmouseout ()<br>window.status = “Ready”<br>End Sub<br>--><br></SCRIPT><br></HTML> |
| x, y | Видає координати покажчика миші, який знаходиться над елементом, який викликав подію. В наступному прикладі в рядок стану виводиться ім’я тега елемента документа, над яким зараз знаходиться покажчик миші.<br><HTML><br><HEAD><br><SCRIPT language = “VBScript” ><br><! –<br>Sub document_onmousemove ()<br>Dim obj<br>Set obj = document.elementFromPoint (window.event.x, window.event.y)<br>window.status = obj.tagName<br>End Sub<br>--><br></SCRIPT><br></HEAD><br><BODY><br><H1>Заголовок </H1><br><P>Параграф </P> |

| | |
|---|---|
| | <BODY><br></HTML> |
| AltKey | Видає True, коли натиснута клавіша <Alt> |
| CtrlKey | Видає True, коли натиснута клавіша <Ctrl> |
| ShiftKey | Видає True, коли натиснута клавіша <Shift> |
| Button | Ідентифікує натиснуту кнопку миші:<br>0 — кнопки не натиснуті;<br>1 — натиснута ліва кнопка;<br>2 — натиснута права кнопка |
| ReturnValue | Коли значення рівне False, то подія відміняється, яку генеруємо елементом за замовчуванням.<br>Приклад відміни переходу по гіперпосиланнях при натиснутій клавіші <Ctrl>:<br><HTML><br><HEAD><br><SCRIPT language = “VBScript” ><br><! –<br>Sub cancelLink ()<br>If window.event.srcElement/tagName = “A” _<br>And window.event.ctrKey Then<br>window.event.returnValue = False<br>End If<br>End Sub<br>--><br></SCRIPT><br></HEAD><br><BODY onclick = “cancelLink ()><br>Зробити перехід до нового <A href = “clock.htm”>документу </A><br></BODY><br></HTML> |

*Змінити колір шрифту і вигляд покажчика миші*

Розглянемо приклад розробки програмного коду для зміни кольору фрази “Visual Basic” та її фону на червоний і жовтий відповідно в той час, коли покажчик миші буде знаходитися над нею, а тип покажчика миші зміниться на руку з вказівним пальцем. Для цього ідентифікуємо слова “Visual Basic” за допомогою тегів <SPAN> і встанлвлення значень їх атрибутів id. Потім треба зробити напис процедури, що обробляє подію onMouseMove об’єкта document. В процедурі спочатку визначається елемент документа, над яким знаходиться покажчик миші. Наприклад:

```
Dim Target
Set Target = window.event.srcElement
```

де властивість srcElement об’єкта event повертає елемент в документ, який викликав генерацію даної події. За допомогою властивостей color і backround об’єкта style зостається тільки встановити кольори і тип покажчика миші. Аналогічно йде обробка події onMouseOut об’єкта document.

Тип покажчика миші встановлюється за допомогою властивості cursor об’єкта style. Ця властивість має такі доступні значення.

| **Значення** | **Опис** |
|---|---|
| Auto | Броузер вибирає тип курсору сам в залежності від контексту |
| Crosshair | Перехрестя |
| Default | Використання в системі за замовчуванням |
| Hand | Рука |
| Move | Маркер переміщення |
| e-resize, ne-resize, nw-resize, n-resize, se-resize, sw-resize, s-resize, w-resize | Стрілка, що вказує в даному напрямку |

| | |
|---|---|
| Text | \| — маркер |
| Wait | Піщані часи |
| Help | Стрілка зі знаком питання |

Програмний код:

```
<HTML>
<HEAD><TITLE>Виділення кольором тексту </TITLE>
<SCRIPT language = “VBScript” >
<! –
Sub Document_OnMouseMove ()
Dim Target
Set Target = window.event.srcElement
‘Об’єкт, на котрому знаходиться курсор
If Left (Target.id, 5) = “spnVB” Then
‘Ідентифікація об’єкта по його id
Target.style.color =”Red”                    ‘Колір шрифту
Target.style.background = “Yellow”           ‘Колір фону
Target.style.fontStyle = “Italic”            ‘Курсивний шрифт
Target.style.fontSize = “110%”               ‘Розмір шрифту
Target.style.cursor = “hand”                 ‘Курсор — рука
End If
End Sub
Sub Document_OnMouseOut ()
Dim Target
Set Target = window.event.srcElement
‘Об’єкт, на котрому знаходиться курсор
If Left (Target.id, 5) = “spnVB” Then
‘Ідентифікація об’єкта по його id
Target.style.color =”Black”                  ‘Колір шрифту
Target.style.background = “White”            ‘Колір фону
Target.style.fontStyle = “Normal”            ‘Звичайний шрифт
Target.style.fontSize = “110%”               ‘Розмір шрифту
Target.style.cursor = “default”              ‘Курсор, що використовується
                                              ‘по замовчуванню
```

```
End If
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<SPAN id = “spnVB1”>Visual Basic</SPAN> — це сила
<SPAN id = “spnVB2”>Visual Basic</SPAN> — це миогутьність
</BODY>
</HTML>
```

*Виділення поточного слова*

Розглянемо приклад виділення фрагменту (в даному випадку, будь-якого слова) при переміщенні над ним покажчика миші. Крім того, виділене слово буде відображатися в рядку стану. Для цього застосуємо метод createTextRange (), повертаючи об’єкт TextRange, який подається як текст із HTML-елемента. Ще можна використовувати наступні методи об’єкта TextRange.

| **Метод** | **Опис** |
|---|---|
| MoveToPoint | Переміщує об’єкт TextRange до точки з вказаними координатами. Синтаксис: MoveToPoint (x, y) |
| Expand | Виділяє діапазон тексту в залежності від значення параметра *unit*. Синтаксис: Expand (*unit*)<br>Параметр *unit* приймає такі допустимі значення: character (символ), word (слово), sentence (речення), textedit (цілий діапазон) |
| Select () | Виділяє діапазон |
| Text | Повертає текст, що міститься у діапазоні |

Програмний код:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
```

```
<! –
Sub Document_OnMouseMove ()
Dim rng
Set rng = window.document.body.createTextRange ()
rng.expand (“word”)
rng.select ()
window.status = rng.text
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<H2>Цитата.</H2>
<P>Зроби діло, гуляй сміло!</P>
</BODY>
</HTML>
```

### 13. 2. 5. Оператор eval

Оператор *eval* перетворює в об’єкт або значення. Наприклад, в наступному коді в обох діалогових вікнах відображається значення 3.

```
Dim Res
Res = “1 + 2”
alert eval (Res)
alert eval (1 + 2)
```

### 13. 2. 6. Вибір зображення

Наступний код демонструє, як за допомогою обробки події onClick рядка сторінки (малюнка) здійснюється вибір одного з двох малюнків. Вибір малюнка здійснюється при клацанні на рядку таблиці, в якому знаходиться малюнок.

Програмний код:

```
<HTML>
<HEAD>
```

```
<TITLE>Вибір малюнка</TITLE>
<SCRIPT language = “VBScript” for = “window” event = “Onload” >
<! - -
imgClub.style.borderColor =”yellow”
imgHeart.style.borderColor = “yellow”
- - >
<SCRIPT language = “VBScript” >
<! - -
Sub trClub_OnClick ()
      imgClub.border = 2
      imgHeart.border = 0
      tdClub.bgColor = “white”
      tdHeart.bgColor = “yellow”
End Sub
Sub trHeart_OnClick ()
      imgClub.border = 2
      imgHeart.border = 0
      tdClub.bgColor = “white”
      tdHeart.bgColor = “yellow”
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<TABLE border = 1>
<TR id = “trClub”>
<TD><IMG id = “imgHeart” src = “Heart.gif”></TD>
<TD id = “tdHeart”>Черві </TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

Наступний код за допомогою обробки події onMouseOver і onMouseOut змінює покажчик миші, який знаходиться над малюнком, а після виходу за межі малюнку повертає покажчику миші початковий вигляд.

Програмний код:

```
<IMG id = “imgMy” src = “Heart.gif” onMouseOver = “”imgMy.src = ‘club.gif’” onMouseOut = “imgMy.src =’Heart.gif’”>
```

### 13. 2. 7. Зміни фрагмента тексту

В документі допустимо змінювати його фрагменти в залежності від дій користувача. Для цього застосовують наступні властивості відповідного елемента документа:

| **Властивість** | **Опис** |
|---|---|
| innerText | Змінює текст, який розташований між парними тегами |
| outerText | Змінює як текст, який розташований між парними тегами, так і самі теги |
| innerHTML | Змінює HTML-код, розташований між парними тегами |
| outerHTML | Змінює як HTML-код, розташований між парними тегами, так і самі теги |

Розглянемо приклад. В вікні броузера є надпис:

Створювали Ви раніше код на VBScript? Так

Слово “Так” має зелений шрифт і жовтий фон. При розміщенні курсора над цим словом курсор приймає вигляд руки. При клацанні на слові “Так” воно змінює своє значення на слово “Ні” червоного кольору. І навпаки, при клацанні по слову “Ні”, на екрані відображається слово “Так” зеленого кольору.

Програмний код:

```
<HTML>
<HEAD>
<TITLE>Зміна фрагмента тексту </TITLE>
<SCRIPT language = “VBScript” >
<! - -
```

```
Sub spaQuiry_OnClick ()
 Select Case Trim(spaQuiry.innerText)
      Case “Tak”
            spaQuir.innerText = “Hi”
            spaQuir.style.color = “red”
      Case “Hi”
            spaQuir.innerText = “Tak”
            spaQuir.style.color = “green”
 End Select
End Sub
Sub spaQuir_OnMouseMove ()
 spaQuir.style.cursor = “hand”
End Sub
Sub spaQuir_OnMouseOut ()
      spaQuir.style.cursor = “default”
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
```

Створювали Ви раніше код на VBScript?

```
<SPAN style = “background: yellow; color: green” id = “spaQuiry”> Tak
</SPAN>
</BODY>
</HTML>
```

Коли при переході від слова “Hi” до слова “Tak” ми бажали, щоб шрифт встановлювався напівжирним, підкресленим, а навпаки встановлювався нормальним, то в цьому випадку треба використовувати властивість innerHTML, це подано в наступному програмному коді.

```
Sub spaQuiry_OnClick ()
 Select Case Trim(spaQuiry.innerText)
      Case “Tak”
            spaQuir.innerHTML = “<B><U>Hi</U> </B>”
```

```
            spaQuir.style.color = “red”
        Case “Hi”
            spaQuir.innerHTML = “Tak”
            spaQuir.style.color = “green”
 End Select
End Sub
```

*Електронний годинник*

За допомогою тега <SPAN> в документі можна відмітити те місце, де буде відображатися час, включити таймер і відобразити поточне значення таймера у вибраному місці. Розглянемо такий приклад.

Програмний код:

```
<HTML>
<SCRIPT language = “VBScript” >
<! - -
Sub window_On Load ()
      Window.setInterval “Timer (), 500
End Sub
Sub Timer ()
      trm.innerText = time ()
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<P>Поточний час <SPAN style = “font-size: larger; color: red;
                  background-color: yellow” id = “trm”> </SPAN></P>
</BODY>
</HTML>
```

*Зміст фрейма, що динамічно змінюється*

Демонстраційний проект складається з двох файлів.

| **Файл** | **Опис** |
|---|---|
| FrameD.htm | Розбиває по вертикалі вікно броузера на два фрейми |
| MenuFruit.htm | Виводить в лівий фрейм меню вибору товарів: фрукти або овочі. Вибір елемента приводить до того, що в правому фреймі (його ім'я frashow) відображається в таблиці зі списком товарів вибраної категорії. Таблиця створюється в коді, в якому спочатку треба створити об'єкт, що відповідає документу, який виводиться в конкретний фрейм (в нашому випадку, frashow).<br>Dim fra<br>Set fra = parent.frames ("frashow"), document<br>Зверніть увагу, що в лівому фреймі зі списком вибору не має реальних гіперпосилань, а тільки їх імітація. Імітація забезпечується двояким способом: текст виводиться підкресленим, а у вибраному елементі змінюється шрифт. |

Програмний код із файла FrameD.htm:

```
<HTML>
<FRAMESET COLS  = "30%, 70%">
<FRAME SRC =="MenuFruit.htm" NAME = "fraMenu">
< FRAME NAME  = "frasShow">
</ FRAMESET >
</HTML>
```

Програмний код із файла MenuFruit.htm:

```
<HTML>
<SCRIPT language = "VBScript" >
<! - -
Sub ShowFruits ()
 document.all ("aFruits").style.color = "Navy"
 document.all ("aVegetables").style.color = "Black"
```

```
  Dim fra
  Set fra = parent.frames (“fraShow”).document
  fra.open
  fra.write “<TABLE border = 1 >”
  fra.write “<TR>”
  fra.write “<TD> Апельсини</TD>”
  fra.write “<TD>18 p.</TD>”
  fra.write “</TR>”
  fra.write “<TR>”
  fra.write “<TD Яблука</TD>”
  fra.write “<TD>18 p.</TD>”
  fra.write “</TD>”
  fra.write “</TR>”
  fra.write “</TABLE>”
  fra.close
  Set fra = Nothing
End Sub
Sub aFruits_onmouseover ()
  aFruits.style.cursor = “hand”
End Sub
Sub aFruits_onmouseout ()
  aFruits.style.cursor = “default”
End Sub
Sub ShowVegetables ()
  document.all (“aFruits”).style.color = “Black”
  document.all (“aVegetables”).style.color = “Navy”
  Dim fra
  Set fra = parent.frames (“fraShow”).document
  fra.open
  fra.write “<TABLE border = 1 >”
  fra.write “<TR>”
  fra.write “<TD> Морква</TD>”
  fra.write “<TD>18 p.</TD>”
```

```
 fra.write “</TR>”
 fra.write “<TR>”
 fra.write “<TD Картопля</TD>”
 fra.write “<TD>18 p.</TD>”
 fra.write “</TD>”
 fra.write “</TR>”
 fra.write “</TABLE>”
 fra.close
 Set fra = Nothing
End Sub
Sub aVegetables_onmouseover ()
 aVegetables.style.cursor = “hand”
End Sub
Sub aVegetables_onmouseout ()
 aVegetables.style.cursor = “default”
End Sub
-->
</SCRIPT>
<BODY>
<H2>Ваш вибір</H2>
<OL>
 <L1> <A id = “aFruits” onClick = “ShowFruits () > <U>Овочі</U>
 <L1>  <A  id  =  “aVegetables”  onClick  =  “ShowVegetables  ()  >
Фрукти<U>Овочі</U>
</OL>
</BODY>
</HTML>
```

### 13. 2. 8. Об’єкт screen

Об’єкт *screen* видає інформацію про екран користувача. Перерахуємо основні властивості цього об’єкту.

| Властивість | Опис |
|---|---|
| height, width | Висота і ширина екрана в пікселях |
| availHeight, availWidth | Висота і ширина, які доступні для вікна броузера |
| colorDepth | Число бітів на піксель, що використовуються екраном |
| updateInterval | Задає інтервал в мілісекундах, через який оновлюється зміст екрану |

Наступний код, в залежності від значення властивості colorDepth, створює або чорно-білий, або кольоровий документ.

Програмний код визначення типу екрана:

```
<HTML>
<SCRIPT language = “VBScript” >
<! - -
Sub window_OnLoad ()
    If screen.colorDepth > 2 then
       document.bgColor = “lime”
       document.bgColor = “blue”
    Else
       document.bgColor = “black”
       document.bgColor = “white”
    End If
End Sub
-->
</SCRIPT>
<BODY>
<H1>Тест</H1>
</BODY>
</HTML>
```

### 13. 2. 9. Динамічні фільтри

Фільтри дозволяють створювати більшу виразність документам. Динамічний фільтр можна створювати за допомогою властивості filters, яка надає ціле

сімейство всіх фільтрів. Далі треба вказати ім'я фільтру, після чого можна встановити його властивість.

В наступному прикладі створюється ореол, що динамічно змінюється навколо тексту.

Програмний код ореола:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<! - -
Public tmr, iCount, iStep
iCount = 0
iStep = 1
Sub window_OnLoad ()
      tmr=window.setInterval(“MyFilter(), 100)
End Sub
Sub MyFilter()
 hName.filter.Glow.strength=iCount
 iCount = iCount + iStep
 If iCount >= 10 Or iCount <= Then iStep = -iStep
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<DIV  id=”hName” style=”font: 40pt bold; color: green;
                          position: absolute; top: 0;
                          filter: Glow(Color #FF0000, strength)”>
Andrey Micho
</DIV>
</BODY>
</HTML>
```

Створимо ефект тіні, що обертається навколо зображення оригіналу.

Програмний код тіні.

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<! - -
Public Fi
Const Pi = 3.1415926
Fi = 0
Sub window_OnLoad ()
      window.setInterval “MyFilter(), 100
End Sub
Sub MyFilter()
 hName.filters.DropShadow.OffX =10*Cos(2*Pi*Fi/360)
 hName.filters.DropShadow.OffY =10*Cos(2*Pi*Fi/360)
 Fi =Fi + 10
 If Fi >= 360 Then Fi = 0
End Sub
-->
</SCRIPT>
</HEAD>
<BODY bgcolor=black>
<DIV id=”hName” style=”font: 40pt bold; color: red; position: absolute;
top: 0; filter: DropShadow(Color=aqua, OffX=10, OffY=0)”>
Andrea Macho
</DIV>
</BODY>
</HTML>
```

### 13. 2. 10. Створення ефекту переходу

Параметр filter атрибута STYLE дозволяє реалізувати візуальні спецефекти. Ефект переходу створюється, коли вказати як значення цього параметра revealtrans.

{filter: revealtrans(duration=*duration,* transition=*transitionshape* )}

- *duration* — тимчасовий інтервал ефекту переходу, який має формат seconds.milliseconds(0.000);
- *transitionshape* — ціле число від 0 до 23, яке задає перехід.

| **Константа** | **Тип переходу** |
|---|---|
| 0 | Box in (прямокутник всередині) |
| 1 | Box out (прямокутник зовні) |
| 2 | Circle in (коло всередині) |
| 3 | Circle out (коло зовні) |
| 4 | Wipe up (стирання вгору) |
| 5 | Wipe down (стирання вниз) |
| 6 | Wipe right (стирання праворуч) |
| 7 | Wipe left (стирання ліворуч) |
| 8 | Vertical blinds (вертикальні жалюзі) |
| 9 | Horizontal blinds (горизонтальні жалюзі) |
| 10 | Checkerboard across (клітки поперек) |
| 11 | Checkerboard down (клітки вниз) |
| 12 | Random dissolve (випадкове розчинення) |
| 13 | Split vertical in (замикаючі штори по вертикалі) |
| 14 | Split vertical out (відкриваючі штори по вертикалі) |
| 15 | Split horizontal in (замикаючі штори по горизонталі) |
| 16 | Split horizontal out (відкриваючі штори по горизонталі) |
| 17 | Strips left down (кутом вліво вниз) |
| 18 | Strips left up (кутом вліво вгору) |
| 19 | Strips right down (кутом вправо вниз) |
| 20 | Strips right up (кутом вправо вгору) |
| 21 | Random bars horizontal (випадкові горизонтальні смуги) |
| 22 | Random bars vertical (випадкові вертикальні смуги) |
| 23 | Random (випадковим чином) |

Розглянемо приклад. У документі вибрана область, в якій виводиться то “Macho” на зеленому фоні, то “Andrea ” на жовтому фоні. Перехід від одного напису до другого виконується за допомогою створення ефекту переходу, причому кожний раз новим. Перерахуємо методи об’єкта filter, які керують процесом виконання фільтра.

| **Метод** | **Опис** |
|---|---|
| play | Грає фільтр. Синтаксис:<br>play(duration)<br>де duration – час, на протязі якого здійснюється ефект переходу. Коли значення параметра не вказується, гра виконується на протязі часу, який вказаний значенням параметра duration у revealtrans атрибута STYLE |
| apply | Застосовується фільтр |

Програмний код типів поступового заміщення одного тексту іншим:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
Sub init ()
      setTimeout “start ()”, 100
End Sub
Sub start ()
      Dim iCount
      iCount = divMy.filters (0) .Transition
      If iCount = 23 Then
            divMy.filters (0).Transition = 1
      Else
            divMy.filters (0).Transition = iCount + 1
      End If
      divMy.filters (0).Apply ()
      If divMy.innerText = “Andrea ” Then
```

```
            divMy.style.backgroundColor = “green”
            divMy.innerText  = “Macho”
      Else
            divMy.innerText = “Andrea ”
            divMy.style.backgroundColor = “yellow”
      End If
      divMy.filters (0).Play ()
End Sub
Sub done
      start
End Sub
</SCRIPT>
</HEAD>
<BODY onload=”init ()”  bgcolor =”black”>
<DIV  id=”divMy” style=”position: absolute; top: 20px;left: 20px;
width: 400px; height: 100px; font-size: 100; text-align: center; background-
color:  blue;  filter:  revealTrans(Transition=1,  Duration=1.5)”
onfilterchange=”done ()”>
Macho
</DIV>
</BODY>
</HTML>
```

Розглянемо приклад, у якому змінюються не переходи, а відображається мій текст. В заданій області послідовно розчинюються один в одному написи.

Програмний код:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
      Public TextOut (2)
      TextOut (0) = “Андрей Махно”
      TextOut (1) = “Excel, VBA, Internet”
      TextOut (2) =  “в економіці і фінансах”
      Sub Window_onload ()
```

```
            setInterval “showText ()” , 3000
        End Sub
 Dim I
 I = -1
Sub showText ()
 If I = 2 Then
        I = 0
 Else
        I = I + 1
 End If
 div1.filters (0).apply ()
 div1.innerText = TextOut (I)
 div1.filters (0).apply ()
        End Sub
</SCRIPT>
</HEAD>
<BODY bgcolor =black onload=”showText ()”>
<FONT color=yellow>
<DIV  id=”div1” style=”position: absolute; top: 20;left: 20;
width: 300; height: 300; font-size: 40; text-align: center;
 filter: revealTrans(Transition=12, Duration=2)”>
</DIV>
</BODY>
</HTML>
```

## 13.3. Питання для самоконтролю

1. Для чого використовується мова VBScript?
2. Що дозволяє робити мова VBScript ?
3. Які команди треба виконати для виклику Microsoft Development Environment в Microsoft FrontPage ?
4. Де розташовується код VBScript ?
5. Що вказує атрибут LANGUAGE тега <SCRIPT>?

6. Який єдиний тип даних є в мові VBScript ?
7. Якими операторами об’являються змінні в мові VBScript?
8. Яка об’єктна модель використовується при написанні сценаріїв ?
9. Об’єкт *window*?
10. Об’єкт *document*?
11. Що можна зробити властивістю length?
12. Що виконують властивостями: left, top, height, width об’єкта *style* ?
13. У якому сценарію здійснюється ініціалізація змінних рівня модуля ?
14. Об’єкт *navigator*?
15. Об’єкт *event*?
16. Оператор *eval*?
17. Об’єкт *screen*?
18. Динамічні фільтри?
19. Як створити ефект переходу?

## 14. ФОРМИ І ЕЛЕМЕНТИ КЕРУВАННЯ

Інформація швидко старіє. Тому нас приваблюють можливості Web-документів, які дозволяють динамічний обмін між користувачем і сервером. Одним з таких засобів є HTML-форми і CGI-(Common Gateway Interface, Загальний інтерфейс шлюзів) і VBScript-сценарії. В HTML форма застосовується для введення даних і відправки повідомлень серверу. Синтаксис:

```
<FORM
    ACTION=url
    CLASS=classname
    ENCTYPE=encoding
    ID=value
    LANG=language
    LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
    METHOD=GET | POST
    NAME=name
```

STYLE=*cssl-properties*

TARGET=*window_name* | _blank | _parent | _self | _top

TITLE =*text*

*event* = *script*

>

Розглянемо опис атрибутів, які часто використовуються.

□ NAME — ім'я форми для наступних посилань на неї;

□ ACTION — URL процес на сервері, який отримує дані із форми;

□ METHOD — задає метод передачі даних від клієнта серверу. Доступні значення:

- GET — цей метод використовується за замовчуванням і приводить до додавання змісту форми до URL. Потім сервер передає дані в програму CGI як аргумент командного рядка, який ця програма може обробити.
- POST — при використані цього метода зміст форми пересилається не як частина URL, а у вигляді окремого файла. Сервер потім передає в програму CGI, з якого в міру необхідності списуються дані. Рекомендується застосовувати цей метод при великих обсягах інформації.

Між парами тегів <FORM>…</FORM> можна розташовувати теги, що задають елементи управління, інші HTML-елементи, текст, а також форми Web-сторінок.

## 14. 1. Поле уведення, кнопка, прапорець і перемикач

Елементи управління на формі створюються тегом <INPUT>. Цей тег дозволяє створювати різні елементи управління. Синтаксис:

<INPUT

ACCESSKEY=*key*

ALIGN=LEFT | CENTER | RIGHT

ALT=*text*

CLASS=*classname*

DISABLED

DYNSRC=*url*

```
 ID=value
 LANG=language
 LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
 LOWSRC=url
 MAXLENGTH=n
 NAME=name
 READONLY
 SIZE=n
 SRC=url
 STYLE=cssl-properties
 TABINDEX=n
 TITLE=text
 TYPE=BUTTON | CHECKBOX | FILE | HIDDEN | IMAGE | PASSWORD |
      RADIO | RESET | SUBMIT | TEXT
 VALUE=value
 event = script
>
```

□ ALIGN — задає вирівнювання тексту, що розташований за елементом управління.

□ CHECKED — прапорець або перемикач, встановлений при завантаженні сторінки. Атрибут може бути вказаний, коли атрибут TYPE має значення CHECKBOX або RADIO.

□ DISABLED — елемент управління недоступний користувачу має сірий колір.

□ MAXLENGTH — максимальна кількість символів, які можуть бути введенні в поле введення або пароль.

□ NAME — ім'я елемента управління, за яким на нього в наступному посилаються в коді сценарію.

□ SIZE — задає розмір поля введення або пароля в символах. По замовчуванню має значення 20. У багаторядкових полів введення задає довжину і висоту.

□ SRC — адреса зображення, коли значення атрибута TYPE дорівнює IMAGE.

□ TYPE — тип елемента управління. Допустимі значення:

• BUTTON — кнопка. Атрибут VALUE задає текст, який зображається на кнопці;

• CHECKBOX — прапорець. Атрибут VALUE визначає стан прапорця: on (встановлений), off (знятий). Атрибут CHECKED вказує, що прапорець встановлено при завантаженні сторінки;

• HIDDEN — сховане значення. Застосовується для відсилання даних серверу. Атрибут VALUE визначає значення елемента;

• PASSWORD — поле введення пароля, у якому введені символи замінюються зірочками. Атрибут VALUE визначає значення елемента управління за замовчуванням. Атрибут MAXLENGTH задає максимальну кількість символів пароля. Атрибут SIZE задає довжину елемента управління в символах;

• IMAGE — малюнок;

• RADIO — перемикач. Атрибут VALUE визначає стан перемикача: on (встановлений), off (знятий). Атрибут CHECKED вказує, що прапорець *перемикач* встановлено при завантаженні сторінки;

• RESET — кнопка, яка встановлює у всіх інтерфейсних елементах значення, які використовуються за замовчуванням;

• SUBMIT — кнопка, дія якої зводиться до відсилання змісту заповненої форми на сервер;

• TEXT — поле введення.

□ TITLE —спливна підказка.

□ VALUE — значення елемента управління.

□ *event* — одна із наступних подій:

| | | | |
|---|---|---|---|
| onAfterUpdate | onDragStart | onKeyUp | onKeyDown |
| onBeforeUpdat | onDblClick | onMouseDown | onSelect |
| OnDlur | onFocus | onMouseMove | onSelectStart |
| OnChange | onHelp | onMouseOver | |
| OnClick | onKeyPress | onMouseUp | |

*Примітка:*

Події в HTML і VBScript мають такі ж назви, що і в VBA, за винятком того, що їх назви починаються з on. Новим є onBlur, що відповідає події, яка генерується елементом управління при втраті ним фокусу.

*Кнопка*

Створимо на Web-сторінці кнопку, при натисненні на яку на екрані з'явиться зображення "Hello, Word!".

Програмний код такий:

```
<HTML>
<HEAD>
<SCRIPT language = "VBScript" >
<!--
Sub cmdHi_onClick
    alert "Hello, Word!", vbExclamation
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
    <INPUT type="button" value="Hello" id="cmdHi">
</FORM>
</BODY>
</HTML>
```

В другому прикладі при створенні кнопки вказується подія оброблення і зв'язана з нею процедура в тезі <INPUT>.

```
<HTML>
<HEAD>
<TITLE> Hello </TITLE>
<SCRIPT language = "VBScript" >
<!--
Sub Hi ()
    alert "Hello, Word!", vbExclamation
```

```
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
    <INPUT type="button" value="Hello" onClick="Hi ()">
</FORM>
</BODY>
</HTML>
```

В третьому прикладі зв'яжемо процедуру обробки події onClick не за рахунок значення атрибутів тега <INPUT>, а тега <SCRIPT>. В цьому випадку нема потреби застосовувати ключове слово Sub.

```
<HTML>
<HEAD>
<TITLE> Hello </TITLE>
<SCRIPT language = "VBScript" for="cmdHi" event="OnClick" >
<!--
    alert "Hello, Word!"
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
    <INPUT name="cmdHi" type="button" value="Hello" id="cmdHi">
</FORM>
</BODY>
</HTML>
```

*Спливаюча підказка*

Спливаюча підказка задається за допомогою встановлення значення атрибута TITLE. В наступному прикладі у кнопці встановлюється спливаюча підказка "Це кнопка ОК". Для того, щоб кнопка **ОК** не була дуже маленькою,

при встановлені значення її атрибута VALUE слово “**ОК**” додатково оточують пропусками.

```
<HTML>
<BODY>
<INPUT name=”cmdOK” type=”button” value=” OK ” title=”Ця кнопка ОК”>
</BODY>
</HTML>
```

*Оживлення кнопки*

За допомогою обробки подій onMouseDown і onMouseUp можна при натиснутій кнопці зробити її колір світло-голубим та на поверхні кнопки “Down” зробити напис червоним шрифтом. Відпущена кнопка має шрифт і колір фону, який встановлено за замовчуванням, з написом слова “Up”.

Програмний код:

```
<HTML>
<HEAD>
<STYLE type=”text/css”>
      .Down{background: aqua; color: red; font-weight: bold}
</STYLE>
<!--светло-голубой фон, красный, полужирный шрифт -->
<SCRIPT language = “VBScript” >
<!—
Sub cmdTest_OnMouseDown ()
      cmdTest.value=”Down”
      cmdTest.className=”Down”
End Sub
Sub cmdTest_OnMouseUp ()
      cmdTest.value=”Up”
      cmdTest.className=”” ‘ Стиль, використовує мій за замовчуванням
End Sub
-->
</SCRIPT>
</HEAD>
```

```
<BODY>
<INPUT name="cmdTest" type="button" value=" Up ">
</BODY>
</HTML>
```

*Різнокольорові кнопки*

Кнопки можна розфарбувати -як їх поверхні, так і розташовані на них слова. Для цього застосовують парний тег <BUTTON>. В наступному коді на кнопці зображається напис "Замовлення ЗАРАЗ!". Перше слово напису синього кольору, а друге — червоного. Крім того, друге слово має курсивний шрифт, розмір якого збільшений по відношенню до першого слова.

```
<HTML>
<BODY>
<BUTTON name="Order" style="font-size: medium; color: blue">
Замовлення
<SPAN style="font-size: large; font-style: italic; color: red">
ЗАРАЗ!</SPAN>
</BUTTON>
</BODY>
</HTML>
```

*Швидкі клавіші*

За допомогою встановлення значення атрибута ACCESSKEY, який означає літеру, таким чином, щоб процедура, що зв'язана з елементом управління, виконувалася при натисненні клавіш[Alt+літера]. При цьому літра для швидкої клавіші виводиться підкресленою. Тому цю кнопку зручніше створювати тегом <BUTTON>.

Програмний код:

```
<HTML>
<HEAD>
<SCRIPT language = "VBScript" >
<!--
Sub cmdOK_OnClick ()
 alert "Ви натиснули кнопку ОК"
```

```
End Sub
Sub cmdCancel_OnClick ()
 alert “Ви натиснули кнопку Cancel”
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM>
 <BUTTON name=”cmdOK” accesskey=”O”> <U>O</U>K</BUTTON>
 <BR>
<BUTTON name=”cmdCancel” accesskey=”C”>
            <U>C</U>ancel</BUTTON>
</FORM>
</BODY>
</HTML>
```

## 14.2. Поле уведення, прапорець, перемикач

*Поле уведення*

Основною властивістю поля уведення є Value, яка видає або встановлює значення, що відображається в полі уведення. Ця властивість є майже у всіх елементів управління.

Розглянемо приклад використання поля уведення. На сторінці відображаються два елементи управління: кнопка і поле уведення. Користувач заносить в поле уведення ім’я і натискує кнопку. Програма видає користувачу ім’я або інформує, що він забув ввести ім’я, коли зміст поля порожній.

Програмний код приклада:

```
<HTML>
<HEAD>
<TITLE>Вітання</TITLE>
<SCRIPT language = “VBScript” >
<!--
```

```
Sub cmdHi_OnClick ()
 Dim sName
 sName = frmMyForm.txtName.Value
 If sName=" " Then
      alert "Ви забули ввести ім'я", vbExclamation
      frmMyForm.txtName.focus
 Else
      alert "Привіт," & sName, vbExclamation
 End If
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM name="frmMyForm">
      Ввести Ваше ім'я  <INPUT type="text" name="txtName">
      <P>
      <INPUT type="button" value=Вітання name="cmdHi">
      </P>
</FORM>
</BODY>
</HTML>
```

*Прапорець*

Основною властивістю прапорця є Checked, що приймає значення True, коли прапорець встановлений, і False, коли прапорець скинуто. Наприклад, на Web-сторінці зображені три прапорці і поле уведення. Кожному із прапорців відповідає свій фрукт. Користувач встановлює відповідний прапорець для вибору потрібного фрукту, який виводиться у полі уведення. В полі уведення встановлена властивість Disabled зі значенням True для того, щоб користувач не зміг ввести в нього іншу інформацію, крім списку вибраних фруктів.

Програмний код:

```
<HTML>
```

```
<HEAD>
<TITLE>Вибір фруктів</TITLE>
<SCRIPT language = “VBScript” >
<!--
Public mChoice (2, 1)
Const iFruit = 2
mChoice (0, 1)=”Апельсин”
mChoice (1, 1)=”Банан”
mChoice (2, 1)=”Яблуко”
For I = 0 To iFruit
      mChoice (I, 0) = 0
Next
Sub Output ()
      Dim I, iLast, sFruit
      iLast=0
      sFruit=””
 For I = 0 To iFruit
      If mChoice (I, 0) = 1 Then
            sFruit =sFruit&mChoice (I, 1) & “,”
            iLast = iLast + 1
      End If
 Next
 If iLast > 0 Then
      frmFruit.text1.value=Left(sFruit, Len (sFruit) - 1)
 Else
      frmFruit.text1.value=””
 End If
End Sub
Sub checkbox1_OnClick()
 If frmFruit.checkbox1.checked Then
      mChoice (0, 0) = 1
Else
      mChoice (0, 0) = 0
```

```
 End If
      Output
End Sub
Sub checkbox2_OnClick()
 If frmFruit.checkbox2.checked Then
      mChoice (1, 0) = 1
Else
      mChoice (1, 0) = 0
 End If
      Output
End Sub
Sub checkbox3_OnClick()
 If frmFruit.checkbox3.checked Then
      mChoice (2, 0) = 1
Else
      mChoice (2, 0) = 0
 End If
      Output
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM name="frmFruit" event="OnLoad">
      <INPUT name="checkbox1" type="checkbox">Апельсин
      <P><INPUT name="checkbox2" type="checkbox">Банан
      <P><INPUT name="checkbox3" type="checkbox">Яблуко
      <P><INPUT name="text1" size=40></P>
</FORM>
</BODY>
</HTML>
```

*Перемикач*

Розглянемо приклад, у якому на Web-сторінці розташовані три перемикачі і поле уведення. Перемикачі розташовані у таблиці, в якій встановлено жовтий колір. Кожному перемикачу відповідає свій фрукт. Назва вибраного користувачем фрукта відображається в полі уведення.

Програмний код:

```
<HTML>
<HEAD>
<TITLE>Вибір одного із фруктів</TITLE>
<SCRIPT language = “VBScript” >
<!—
Sub radFruit1_OnClick()
      frmFruit.txtFruit.value=”Апельсин”
End Sub
Sub radFruit1_OnClick()
      frmFruit.txtFruit.value=”Банан”
End Sub
Sub radFruit1_OnClick()
      frmFruit.txtFruit.value=”Яблуко”
End Sub
-->
</SCRIPT>
</BODY>
</HEAD>
<BODY>
<FORM name=”frmFruit”>
<TABLE bgcolor=”yellow”>
      <TR>
            <TD><INPUT id=”radFruit1” name=”radFruit” type=”radio”
                        event=”OnClick”></TD>
            <TD><INPUT id=”radFruit2” name=”radFruit” type=”radio”
                        event=”OnClick”></TD>
            <TD><INPUT id=”radFruit3” name=”radFruit” type=”radio”
                        event=”OnClick”></TD>
```

```
            <TD>Яблуко</TD>
        </TR>
</TABLE>
<P>
<INPUT name="txtFruit">
</FORM>
</HTML>
```

*Масив елементів управління*

Для створення масиву елементів в Visual Basic можна використати об'єкт document та його властивість ActiveElement, який повертає активний елемент сторінки. Активним елементом може бути будь-який тег, а не тільки елемент сторінки. Для ідентифікації активного елемента сторінки застосуємо властивість id, що повертає його ідентифікатор. Ця ідея реалізована в наступному програмному коді.

```
<HTML>
<HEAD>
<TITLE>Вибір одного із фруктів</TITLE>
<SCRIPT language = "VBScript" >
<!—
Sub ChooseIt ()
        Select Case document.activeElement.Id
                Case "radFruit1"
                frmFruit.txtFruit.value="Апельсин"
                Case "radFruit2"
                frmFruit.txtFruit.value="Банан"
                Case "radFruit3"
                frmFruit.txtFruit.value="Яблуко"
        End Select
End Sub
-->
</SCRIPT>
```

```
</HEAD>
<BODY>
<FORM name="frmFruit">
<TABLE bgcolor="yellow">
      <TR>
            <TD><INPUT id="radFruit1" name="radFruit" type="radio"
                        OnClick ="ChooseIt ()></TD>
            <TD>Апельсин</TD></TR>
      <TR>
            <TD><INPUT id="radFruit2" name="radFruit" type="radio"
                        OnClick ="ChooseIt ()></TD>
            <TD>Банан</TD></TR>
            <TD><INPUT id="radFruit3" name="radFruit" type="radio"
                        OnClick ="ChooseIt ()></TD>
            <TD>Яблуко</TD>
      </TR>
</TABLE>
<P>
<INPUT name="txtFruit">
</FORM>
</BODY>
</HTML>
```

Розглянемо ще один приклад створення масиву із чотирьох кнопок жовтого кольору. Натиснення на кнопку змінює її колір з жовтого на зелений і навпаки. Ідея цієї програми полягає в тому, що значенням атрибута id кнопок присвоєні значення cmd1, cmd2, cmd3, cmd4. Оператор eval перетворює ці вирази в об'єкти (або значення). Наприклад, при i рівному 2 наступний оператор присвоює об'єктній змінній cmd значення cmd2.

```
Set cmd = eval ("cmd" & i)
```

Програмний код програми "Масив кнопок":

```
<HTML>
<HEAD>
```

```
<SCRIPT language = “VBScript” >
<! - -
Public bCount (3)
For I = 0 to 3
      bCount (I) = False
Next
Sub window_onLoad ()
      Dim I, cmd
      For I = 1 to 4
            Set cmd =eval (“cmd” & I)
            cmd.style.backgroundColor = “yellow”
      Next
End Sub
Sub DoIt ()
      Dim I
      I = Right (document.activeElement.id, 1)
      ChangeCmd (I)
End Sub
Sub ChangeCmd (I)
      Dim cmd
      Set cmd = eval (“cmd” & I)
      bCount (I - 1) = Not bCount (I - 1)
      Select Case bCount (I - 1)
            Case True
                  cmd.style.backgroundColor = “green”
            Case False
                  cmd.style.backgroundColor = “yellow”
      End Select
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
```

```
<BUTTON id="cmd1" OnClick="DoIt ()> Кнопка 1 </BUTTON>
<BUTTON id="cmd2" OnClick="DoIt ()> Кнопка 2 </BUTTON>
<BUTTON id="cmd3" OnClick="DoIt ()> Кнопка 3 </BUTTON>
<BUTTON id="cmd4" OnClick="DoIt ()> Кнопка 4 </BUTTON>
</BODY>
</HTML>
```

*Візуальне групування елементів управління*

Візуальне групування елементів управління досягається розміщенням між парними тегами <FILEDSET>, в результаті чого ці елементи управління будуть оточені рамкою. Заголовок рамки створюється парними тегами <LEGEND>, які йдуть безпосередньо за відкриваючим тегом <FILEDSET>.

В наступному коді два перемикача **Windows 98 i Windows NT** об'єднуються в групу **Operation System**.

```
<HTML>
<BODY>
<FORM id="frm">
  <FILEDSET style="width: 30%">
      <LEGEND>Operation System</LEGEND>
       <TABLE>
         <TR>
           <TD><INPUT type="radio" id="Win98" name="OS"></TD>
           <TD>Windows NT</TD>
         </TR>
       </TABLE>
  </FILEDSET>
<FORM>
</BODY>
</HTML>
```

При створенні форм буває зручно не тільки групувати елементи управління, але і виділяти кольором саму форму, як зроблено в програмному коді створення форми для підписки.

```
<HTML>
```

```
<HEAD>
<TITLE>Підписка на нову версію MS Office</TITLE>
<STYLE>
<!—
BUTTON {font-family: Arial, san-serif; font-weight: bold;}
TD {font-family: Arial, san-serif; font-weight: bold; font-size: 80%;}
TH {font-family: Arial, sans-serif; font-weight: normal; font-size: 100%;}
.DIALOG{background-color: #CCC;}
.UD {text-decoration: underline;}
- ->
</STYLE>
<SCRIPT language=”BVScript”>
<!- -
Dim sLevel
Sub ChooseIt ()
        Select Case document.activeElement.Id
            Case “radBeginner”
                sLevel = “Початківець”
            Case “radProf”
                sLevel = “Досвідчений”
            Case “radExpert”
                sLevel = “Експерт”
      End Select
End Sub
Sub cmdSubsribe_onlick ()
      Dim sMsg
      sMsg = “E-mail ” & frmSubsribe.txtEmail.value & vbCrLf _
      & “Ваш рівень ” sLevel
      alert sMsg
End Sub
-->
</SCRIPT>
</HEAD>
```

```
<BODY bgcolor=”yellow”>
<FORM name=”frmSubsribe”>
<TABLE border=”3” cellpadding=”4” cellspacing=”0”>
  <TR>
    <TD class=”DIALOG”>
      <FIELDSET>
        <LEGEND>Підпишися на нову версію MS Office</LEGEND>
        <TABLE>
          <TR>
            <TD align=”center”>
              <SPAN class=”UD”>E</span>mail адрес:
               <INPUT type=”text” size=”25” id=”txtEmail”>
            </TD>
          </TR>
          <TR>
            <TD valign=”top”>
      <FIELDSET>
        <LEGEND>Ваш рівень</LEGEND>
        <TABLE bordercolor=”#000000”>
          <TR>
            <TD>
              <INPUT type=”redio” id=”radBeginner”
    name=”radLevel” onclick=”ChooseIt ()”></TD>
            <TD nowrap>Початківець</TD>
           </TR>
          <TR>
            <TD>
              <INPUT type=”radio” id=”radProf”
    name=”radLevel” onclick=”ChooseIt ()”></TD>
            <TD nowrap>Досвідчений</TD>
           </TR>
          <TR>
            <TD>
```

```
                    <INPUT type="radio" id="radExpert"
      name="radLevel" onclick="ChooseIt ()"></TD>
                  <TD nowrap>Експерт</TD>
                </TR>
              </TABLE>
            <FIELDSET>
          </TD>
 </TR>
 <TR>
<TR>
      <TD nowrap>Для підписки натисніть кнопку</TD>
      <TD><BUTTON id="cmdSubscribe">Підписка</BUTTON></TD>
 </TR>
</TABLE>
 </FILEDSET>
 </TD>
 </TR>
</TABLE>
<FORM>
</BODY>
</HTML>
```

*Хования і зображення елементів*

Коли властивість visibility об'єкту style має значення visible, то елемент ми бачимо, а коли hidden, то він схований. Можливо також управління за допомогою властивості display об'єкту style. Коли значення цієї властивості дорівнює block, то елемент ми бачимо, а коли none, то він схований. Різниця між застосуванням властивостей visibility і display полягає у тому, що при застосуванні властивості visibility, коли елемент схований, на екрані залишається порожнє місце, яке було зайнято елементом, а коли застосовується властивість display ховається не тільки елемент, але і позиція, що відведена для нього.

Розглянемо приклад застосування властивості visibility. Користувачу пропонується вказати назву фрукта, який йому до вподоби. З цією метою на екрані відображаються три перемикачі: **Апельсини, Банани** і **Інші**. На екрані є поле уведення, в яке користувач може ввести назву потрібного фрукту.

Програмний код:

```
<HTML>
<HEAD>
<STYLE>
<!--
TD {font-family: Arial, san-serif; font-weight: bold; font-size: 80%;}
CAPTION {font-family: Arial, sans-serif; font-weight: bold;
      font-size: 100%; color: navy}
-->
</STYLE>
<SCRIPT language = “VBScript” >
<!--
Sub window_onLoad ()
      ShowIt
End Sub
Sub ShowIt ()
      If rad3.checked Then
            txtMy.style.visibility =”visible”
      Else
            txtMy.style.visibility =”hidden”
      End If
End Sub
-->
</SCRIPT>
</HEAD>
<TABLE width=”30%”>
<CAPTION>Виберіть фрукт, що Вам до вподоби<.CAPTION>
   <TR>
      <TD>Апельсин</TD>
```

```
        <TD><INPUT type=”radio” id=”rad1” name=”rad”
      onclick=”ShowIt ()></TD>
       <TD></TD>
      </TR>
   <TR>
       <TD>Банани</TD>
       <TD><INPUT type=”radio” id=”rad2” name=”rad”
      onclick=”ShowIt()></TD>
       <TD></TD>
</TR>
   <TR>
       <TD>Інші</TD>
       <TD><INPUT type=”radio” id=”rad3” name=”rad”
      onclick=”ShowIt ()></TD>
       <TD><INPUT type=”textbox” id=”txtMy”></TD>
</TR>
</TABLE>
</BODY>
</HTML>
```

Розглянемо приклад застосування властивості display.

```
<HTML>
<STYLE type=”text/css”>
      H1 {color: navy; cursor: hand}
      OL {cursor: hand}
</STYLE>
<HEAD>
<SCRIPT language = “VBScript” >
<!—
Sub ol1_onClick ()
      ol1.style.display = “none”
End Sub
Sub hF_onMouseOver ()
      If oll.style.display = “none” Then
```

```
            hF.style.cursor=”hand”
      Else
            hF.style.cursor=”default”
      End If
End Sub
Sub hF_onClick ()
      ol1.style.display = “block”
End Sub
Sub ol2_onClick ()
      ol2.style.display = “none”
End Sub
Sub hV_onMouseOver ()
      If ol2.style.display = “none” Then
            hV.style.cursor=”hand”
      Else
            hV.style.cursor=”default”
      End If
End Sub
Sub hV_onClick ()
      ol2.style.display = “block”
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<H1 ID=”hF”>фрукти</H1>
<OL id=”ol1”>
      <LI>Апельсини
      <LI>Банани
</OL>
<H1 id=”hV”>Овочі</H!>
<OL id=”ol2”>
      <LI>Картопля
```

```
        <LI>Буряк
</OL>
</BODY>
</HTML>
```

*Створення меню*

В наступному прикладі спочатку на екрані зображається меню, яке складається з двох елементів: **Фрукти** і **Овочі**. Вибір одного із елементів приводить до того, що на екрані відкривається нова панель меню зі списком фруктів або овочів. Вибір елемента з меню, що розкривається, дозволяє відобразити вікно з повідомленням про вибраний елемент.

Програмний код:

```
<HTML>
<HEAD>
<STYLE type=”text/css”>
<!—
        /*Меню*/
        #Menu {float: left; width: 60pt; background: lightgreen; color: Red;
        border: 2px white outset; cursor: default}
        /*Спочатку меню, що розкривається, сховано*/
        #Menu .PopUp {position: absolute; display: none;
        background: lightgreen; border: 2px white outset;
        width: 60pt; margin: 2pt}
        #Menu P {margin-top: Opt; margin-bottom: Opt}
                .over {color: navy; font-weight: bold}
-->
</STYLE>
<SCRIPT language = “VBScript” >
<!—
Public curPop
curPop = Null
Sub PopUpMenuOff ()
        Menu1.style.display =Empty
```

```
        Menu2.style.display =Empty
End Sub
Sub PopUp ()
        Dim el, elPop
        Set el = window.event.srcElement
        PopUpMenuOff
        If (el.tagName = “P”) and (el.parentElement.id = “Menu”) Then
          Set elpop =document.all (el.sourceIndex + 1)
          elPop.style.pixcelLeft =menu.offsetLeft + menu.offsetWidth - 10
          elPop.style.pixcelTop =el.offsetTop + menu.offsetTop
          elPop.style.display = “block”
          curPop = elPop
        End If
        window.event.cancelBubble = True
End Sub
Sub LightOn ()
        If window.event.toElement.tagName = “P” Then
           window.event.toElement.className = “over”
        End If
End Sub
Sub LightOff ()
        If window.event.fromElement.tagName = “P” Then
           window.event.fromElement.className = Empty
        End If
End Sub
-->
</SCRIPT>
</HEAD>
<BODY onclick=”PopUpMenuOff ()>
<H1>Зроби свій вибір</H1>
<DIV ID=”Menu” OnClick=”PopUp ()
                    OnMouseOver=”LightOn ()
                    OnMouseOut=”LightOff ()>
```

```
<P>Фрукти</P>
<DIV class="PopUp" id="Menu1">
<P OnClick="alert ('Вибрали яблуко')
        OnMouseOver="LightOn ()
        OnMouseOut="LightOff ()>Яблуко</P>
<P OnClick="alert ('Вибрали банан')
        OnMouseOver="LightOn ()
        OnMouseOut="LightOff ()>Банан</P>
<P OnClick="alert ('Вибрали апельсин')
        OnMouseOver="LightOn ()
        OnMouseOut="LightOff ()>Апельсин</P>
</DIV>
<P>Овочі</P>
    <DIV class="PopUp" id="Menu2">
<P OnClick="alert ('Вибрали моркву')
        OnMouseOver="LightOn ()
        OnMouseOut="LightOff ()>Морква</P>
<P OnClick="alert ('Вибрали картоплю')
        OnMouseOver="LightOn ()
        OnMouseOut="LightOff ()>Картопля</P>
</DIV>
</DIV>
</BODY>
</HTML>
```

## 14. 3. Кнопки Submit і Reset

Кнопка ***Reset*** призначена для очищення форми і повернення усіх установок до їх первинного значення за замовчуванням. Її єдиним атрибутом є VALUE, значення якого задає текст, який відображається на кнопці.

Кнопка ***Submit*** призначена для збирання даних з форми і надсилання їх Web-серверу для обробки за URL-адресою, яку задано як значення атрибута ACTION тега <FORM>. Дані з форми на сервер відсилаються у такому форматі:

*Ім'яЕлемента1=Значення1&Ім'яЕлемента2=Значення2...*

Програма на сервері розбиває рядок і здійснює відповідні дії.

Розглянемо приклад передавання на сервер імені та адреси електронної пошти користувача

```
<HTML>
<HEAD>
<TITLE>Передати дані і встановити початкові установки</TITLE>
<STYLE>
<STYLE type="text/css">
<!--
TD {font-family: Arial, san-serif; font-weight: bold;
      font-size: 80;}
.CMD {font-family: Arial, san-serif; font-weight: bold;
      font-style: italic; font-size: 90; color: navy; background-color: aqua}
-->
</STYLE>
<SCRIPT language = "VBScript" >
<!--
Sub txtName_OnBlur
      If frmApp.txtName.value = "" Then
        alert "Забули увести прізвище"
        frmApp.txtName.focus
      End If
End Sub
Sub txtTMail_OnBlur
      If frmApp.TxtEMail.value = "" Then
        alert "Забули увести e-mail"
        frmApp.txtName.focus
      End If
End Sub
-->
</SCRIPT>
</HEAD>
```

```
<BODY>
<FORM id=”frmApp” action=”Radio.htm” method=”post”>
<P>
<TABLE cellPadding=1 cellSpacing=1 width=”75%”>
  <TR>
     <TD>Прізвище</TD>
     <TD><INPUT id=txtEMail”></TD>
  </TR>
  <TR>
     <TD>e-mail</TD>
     <TD><INPUT id=txtEMail”></TD>
  </TR>
</TABLE>
</P>
<INPUT class=”CMD” name=”cmdSubmit” type=”Submit” value=”Submit”>
<INPUT class=”CMD” name=”cmdReset” type=”reset” value=”Reset”>
</FORM>
</HTML>
```

*Примітка*

В даному коді після натиснення кнопки Submit нічого не буде здійснюватися тому, що в якості значення атрибута ACTION не вказана програма, що розташована на сервері, яка обробляє отримані із форми дані.

1. Наступний приклад розглядає створення простішого аналізу і обробки даних, що введенні користувачем. В ньому перевіряються:

   - чи введене ім’я дійсно складається тільки із букв латинського алфавіту;
   - чи введені літа є числом із діапазону від1 до 150;
   - чи присутній в адресі електронної пошти символ @;
   - чи інформується користувач при некоректному введені, і дані пересилаються на сервер тільки після їх виправлення, чи ні.

Програмний код:

```
<HTML>
<HEAD>
```

```
<TITLE>Передати дані і встановити початкові установки</TITLE>
<SCRIPT language = “VBScript” >
<!--
Sub cmdSubmit_onClick ()
      Dim objForm
      Set objForm = Document.frmApp
      Const sLetter=”abcdefghijklmnopqrstuwxyz”
      Dim sName, n, I
      sName=Trim(objForm.txtName.Value)
      n=Len(sName)
      If n=0 Then
        alert “Забули ввести ім’я”
        objForm.txtName.focus
        Exit Sub
      Else
        For I = 1 To n
            If instr(1,sLetter, Lcase(Mid (sName, I, 1)))=0 Then
              alert “Ім’я повино бути набрано латинськими буквами”
              objForm.txtName.focus
              Exit Sub
            End If
  Next
      End If
      Dim iAge
      iAge=Trim(objForm.txtName.Value)
      If IsNumeric (iAge) Then
        If iAge<1 Or iAge>150 Then
            alert “Літа повинні бути від 1 до 150 років”
      objForm.txtName.focus
            Exit Sub
         End If
      Else
            alert “Ввести літа”
```

```
        objForm.txtName.focus
            Exit Sub
        End If
        Dim sEMail
        sEMail=Trim(objForm.txtEMail.Value)
        If sEMail="" Then
            alert "Забули ввести e-mail"
        objForm.txtName.focus
            Exit Sub
         Else
            If Instr(1, sEMail, "@")=0 Then
            alert "В адресі e-mail повинений бути присутній символ @"
        objForm.txtEMail.focus
            Exit Sub
            End If
         End If
        frmApp.submit
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM name="frmApp" action="Radio.htm" method="POST">
<P>
<TABLE cellPadding=1 cellSpacing=1 width="75%">
  <TR>
      <TD>First Name</TD>
      <TD><INPUT name="txtName"></TD>
  </TR>
  <TR>
      <TD>Age</TD>
      <TD><INPUT name="txtAge"></TD>
  </TR>
```

```
  <TR>
      <TD>e-mail</TD>
      <TD><INPUT name=”txtEMail</TABLE>
  </TR>
</TABLE>
</P>
<INPUT name=”cmdSubmit” type=”button” value=”Submit”>
<INPUT name=”cmdReset” type=”button” value=”Reset”>
</FORM>
</BODY>
</HTML>
```

*Примітка*

В програмі ще використовується дія кнопки Reset за допомогою метода reload об’єкта location. Метод reload здійснює перезавантаження всієї сторінки, коли вона була змінена.

*Автоматична корекція даних*

В наступному прикладі в рядку, що введений в поле e-mail, автоматично при натисненні кнопки **ОК** вилучаються усі пропуски.

Програмний код:

```
<HTML>
<HEAD>
<SCRIPT language = “VBScript” >
<!--
Sub cmdOK_onclick ()
      Dim sEMail
      sEMail = txtEMail.value
      sEMail =replace (sEMail, “ “, “”)
      txtEMail.value = sEMail
End Sub
```

```
-->
</SCRIPT>
</HEAD>
<BODY>
E-mail <INPUT  id=”txtEMail”><BR><BR>
<INPUT id=”cmdOK” type=”button” value=”  OK  ”>
</BODY>
</HTML>
```

*Перевірка даних на етапі уведення*

За допомогою обробки події OnKeyPress дані, що вводяться користувачем, можна перевіряти на етапі їх набору. Наступний код дає можливість користувачу в полі вводити тільки числа з інтервалу від 1 до 150.

Програмний код:

```
<HTML>
<!--
<SCRIPT language = “VBScript” >
Sub txtAge_OnKeyPress
      Dim tmp, tmpBut ‘Контроль введення тільки цифри
      tmpBut = window.event.keyCode
      If tmpBut  < 48 Or tmpBut  > 57 Then
        window.event.returnValue = False
      Else
        tmp =CStr  (txtAge.value)
        ‘ Контроль за тим, щоб результат не перевищував 150
        If (tmp & chr(tmpBut) > 150) Then
            window.event.returnValue = False
        End If
      End If
End Sub
-->
</SCRIPT>
<BODY>
```

```
    <LABEL>Age (from 1 to 150)</LABEL>
    <INPUT id="txtAge" type="text" size=3>
</BODY>
</HTML>
```

## 14. 4. Меню або список вибору

Парний тег <SELECT> в багатьох броузерах подається як меню або список вибору. Синтаксис:

```
<SELECT
  ACCESSKEY=key
  ALIGN=ABSBOTTOM | ABSMIDDLE | BASELINE | BOTTOM | LEFT |
      MIDDLE | RIGHT | TEXTTOP | TOP
  CLASS=classname
  DATAFLD=colname
  DATASRC=#ID
  DISABLED
  ID=value
  LANG=language
  LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
  MULTIPLE
  NAME=name
  SIZE=n
  STYLE=css1-properties
  TABINDEX=n
  event = script
>
```

□ MULTIPLE — дозволяє виконати вибір із списку. Наявність MULTIPLE примушує SELECT бути поданим в якості списку вибору незалежно від значення SIZE.

□ NAME — ім'я елемента. Цей атрибут повинен бути тому, що він використовується при надсиланні запиту.

□ SIZE — визначає, яку кількість рядків списку ми бачимо. За замовчуванням число дорівнює 1.

SELECT має парні теги, всередині яких може міститися тільки послідовність тегів <OPTION>, за кожним із яких слідує деяка кількість звичайного тексту (без HTML-виразів), наприклад:

```
<SELECT name=”cars” multiple size=3>
        <OPTION value=1>BMW
        <OPTION value=2>Porsche
        <OPTION value=3 SELECTED>Mercedes
</SELECT>
```

В наступному прикладі в список вводиться перелік фруктів. Після вибору користувачем фрукта із списку на екрані відображається діалогове вікно, яке підтверджує зроблений вибір.

Програмний код роботи зі списком:

```
<HTML>
<HEAD>
<STYLE>
<!—
.LBL{font-family: Arial; san-serif; font-weight: bold;
        font-size: 110%; background-color: aqua; color: red}
.OPT{font-family: Arial; sans-serif; font-weight: bold;
        font-style: italic; font-size: 90%;  color: navy;}
-->
</STYLE>
<SCRIPT language = “VBScript” >
<!--
Sub lstFruit_OnChange
        Dim Index
        Index = lstFruit.lstFruit.SelectedIndex
        alert “Ви вибрали ”& Lcase(lstFruit.lstFruit.Options(Index).Text)
End Sub
-->
```

```
</SCRIPT>
</HEAD>
<BODY>
<FROM id="frmFruit">
<LABEL class="lbl">Виберіть фрукт:</LABEL>
    <SELECT ID="lstFruit" class="opt">
        <OPTION>Апельсин
        <OPTION>Банан
        <OPTION>Яблуко
    </SELECT>
</FORM>
</BODY>
</HTML>
```

*Зображення структури документа*

Розглянемо приклад використання списку вибору. В даному елементі виводяться різні структури документа, які користувач зможе зобразити у вікні броузера: весь документ, тільки заголовок <H1>, тільки заголовки (в наступному випадку <H1>, і <H2). Вибір рівня зображення документа, який має рівень не нижче вибраного, а елементи з більш низьким рівнем ховаються. Вибір створюється в сценарії, причому число елементів списку вибору попередньо не визначено.

Програмний код відображення структури документа:

```
<HTML>
<HEAD>
<STYLE id="all" title="Весь документ" type="text/css">
    #allHead {display: none}
    #h1Head {display: none}
    #allText {color: red; cursor: default}
</STYLE>
<STYLE ID="headers" title="Тільки заголовки" type="text/css" disabled>
    #allText {display: none}
    #h1Head {display: none}
```

```
      #allHead {color:navy; cursor: default}
      DIV {display: none}
</STYLE>
<STYLE ID="headers" title="Тільки заголовки H1"
       type="text/css" disabled>
      #allText {display: none}
      #h1Head {display: none}
      #allHead {color:navy; cursor: default}
      H2, DIV  {display: none}
</STYLE>
<SCRIPT language = VBScript>
Sub lstStyles_OnChange ()
      For iCount = 0 To document.styleSheets.length – 1
            document.styleSheets(iCount).disabled =
                  (lstStyles.selectedIndex <> iCount)
      Next
End Sub
</SCRIPT>
</HEAD>
<BODY>
<H1>Відображення структури документа</H1>
<P>Виберіть рівень, що відображається
<SELECT id="lstStyles">
      <SCRIPT language = "VBScript" >
      For iCount = 0 To document.styleSheets.length – 1
            document.write ("<OPTION>" & _
      document.styleSheets (iCount).title)
      Next
</SCRIPT>
</SELECT>
<P id="allText">
      Відображається весь документ
</P>
```

```
<P id="allHead">
    Відображаються тільки заголовки.
</P>
<P id="h1Head">
    Відображаються тільки заголовки H1.
</P>
<H2>Фрукти</H2>
<DIV>Апельсини, банани, яблука.</DIV>
<H2>Овочі</H2>
<DIV>Морква, картопля, буряк.</DIV>
</BODY>
</HTML>
```

## 14. 5. Багаторядкове поле уведення

Подвійний тег <TEXTAREA> створює багаторядкове поле уведення. Текст, що розташований між початковим і кінцевим тегами, виводиться на екрані при завантаженні документа броузером. Синтаксис:

```
<TEXTAREA
ACCESSKEY=key
  ALIGN=ABSBOTTOM | ABSMIDDLE | BASELINE | BOTTOM | LEFT |
      MIDDLE | RIGHT | TEXTTOP | TOP
  CLASS=classname
  COLS=n
  DATASRC=#ID
  DISABLED
  ID=value
  LANG=language
  LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
  NAME=name
  READONLY
  ROWS=n
  STYLE=css1-properties
  TABINDEX=n
```

TITLE=*text*

WRAP=OFF | PHYSICAL | VIRTUAL

*event = script*

>

□ NAME — ім'я елемента управління.

□ COLS і ROWS — довжина і висота багаторядкового поля уведення в символах.

□ WRAP — визначення переносу слів. Допустимі значення: OFF (не переносити слова, використовується за замовчуванням), PHYSICAL (слова переносяться як на екрані, так і при передаванні даних серверу), VIRTUAL (слова переносяться тільки на екрані, але не при передаванні даних серверу).

Багаторядкове поле введення підтримує події:

| | | | |
|---|---|---|---|
| OnAfterUpdate | OnKeyPress | OnResize | OnKeyDown |
| OnBlur | OnMouseDown | OnScroll | OnKeyUp |
| OnClick | OnMouseUp | OnChange | OnMouseMove |
| OnDragStart | OnRowEnter | OnFocus | OnSelect |

Наприклад:

```
<TEXTAREA COLS=30 ROWS=10>
Подвійний тег &lt;TEXTAREA&gt; створює багаторядкове поле уведення.
</TEXTAREA>
```

## 14.6. Картка-зображення

Картки-зображення (imagemapes) створюються тегами <MAP> і <AREA>.

Парний тег <MAP> має єдиний атрибут Name і є контейнером для тегів <AREA>, які створюють елементи цього набору. Атрибут Name здійснює зв'язок між карткою і конкретним зображенням, що створено за допомогою тега <IMG>. Це здійснюється установкою значення атрибута USEMAP тега <IMG>, який дорівнює значенню атрибута Name тега <MAP>, перед яким треба розташувати символ #. Синтаксис тега <AREA>:

```
<AREA
    ALT=text
    CLASS=classname
```

```
        COORDS=coordinates
        HREF=url
        ID=value
        LANG=language
        LANGUAGE=JAVASCRIPT | JSCRIPT| VBSCRIPT |VBS
        NOHREF
        SHAPE=CIRC | CORCLE | POLYGON | RECT | RECTANGLE
        STYLE=cssl-properties
        TABINDEX=n
        TARGET=window_name | _blank | _parent | _self | _top
        TITLE=text
        event = script
>
```

□ SHAPE — визначає форму чутливої області (картки); значення цього атрибута визначає суть координат, які використовуються як значення атрибута COORDS.

□ COORDS — в цьому атрибуті через кому перераховуються пари координат, які задають чутливу область.

□ HREF — URL посилання.

□ NOHREF — визначає, що при натисненні на чутливу область не здійснюється перехід по посиланню.

Побудуємо картку-зображення у вигляді сторінки, на якій маємо зображення кімнати. При натисненні на каміні або стільці на цьому зображенні на екрані з'являється відповідне повідомлення.

```
<HTML>
<HEAD>
<SCRIPT language = "VBScript" >
<! --
Sub ShowIt ()
        Select Case Right (window.event.srcElement.id, 1)
                Case 1
                        alert   "Камін"
```

```
            Case 2
                alert  “Свічка”
            Case 3
                alert  “Стілець”
        End Select
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<IMG id=”mapImage” src=”back.gif” border=0
    width=311 height=198 usemap=”#Room”>
<MAP name=”Room”>
<AREA shape=”polygon” id=”are1” onclick=”ShowIt ()
    coords=”5, 10, 125, 7, 126, 125, 2, 127”>
<AREA shape=”polygon” id=”are2” onclick=”ShowIt ()
    coords=”109, 169, 136, 163, 137, 69, 145, 68, 148, 162,
     178, 176, 172, 187, 148, 193, 122, 190, 106, 177” >
<AREA shape=”polygon” id=”are3” onclick=”ShowIt ()
    coords=”164, 154, 166, 121, 197, 110, 201, 44, 254, 42, 303,
 54, 306, 181, 243, 198, 180, 156, 168, 156”>
</MAP>
</BODY>
</HTML>
```

*Примітка:*

Для того, щоб автоматизувати процес визначення координат вершин ламаної лінії в пікселях, утворюючи контур зображення, можна скористатися наступною програмою на Visual Basic 6.0. Ця програма на формі відображає потрібний малюнок. Вам тільки залишається на цьому малюнку натиснути мишею вершини ламаної, яку будуємо, а ваші дії програма перетворить в послідовність координат, які шукаються.

Програма автоматизованого побудування координат ламаної лінії:

```
Private Sub Form_Load ()
    Me.AutoRedraw = True
    Me.PaintPicture LoadPicture (App.Path & “\Back.gif”), 0 0
End Sub
Private Sub Form_MouseDown (Button As Integer, Shift As Integer, _
    X As Single, Y As Single)
    Debug.Print X / Screen.TwipsPerPixelX & “,” _
    & Y / Screen.TwipsPerPixelY & “,” ;
End Sub
```

## 14.7. Біжучий рядок

Для створення біжучого рядка в HTML застосовується парний тег <MARQUEE>. Синтаксис:

```
<MARQUEE
    BENAVIOR=ALTTERNATE | SCROLL | SLIDE
    BGCOLOR=color
    CLASS=classname
    DATAFLD=colname
    DATAFORMATAS=HTML | TEXT
    DATASRC=#ID
    DIRECTION=DOWN | LEFT | RIGHT | UP
    HEIGHT=n
    HSPACE=n
    ID=value
    LANG=language
    LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS
    LOOP=n
    SCROLLAMOUNT=n
    SCROLLDELAY=milliseconds
    STYLE=cssl-properties
    TITLE=text
    TRUESPEED
    VSPACE=n
```

WIDTH=*n*

*event = script*

>

□ BENAVIOR — задає тип руху в напрямку, що вказаний значенням атрибуту DIRECTION. Допустимі значення: SCROLL (по досягненню межі екрана процес повторюється), SLIDE (по досягненню межі екрана процес зупиняється).

□ DIRECTION — задає напрям руху біжучого рядка.

□ LOOP — число повторень.

□ SCROLLDELAY — число пікселів, на яке переміщується за один рух біжучий рядок.

□ MARQUEE — швидкість в мілісекундах біжучого рядка.

Наприклад,

```
<MARQUEE direction=left behavior=slide scrollamount=2
    scrolldelay=60 loop=3>
  Andrey Michno
</MARQUEE>
```

Наступний код створює біжучий вгору рядок. Рядок займає тільки частину екрана (розміщену в клітинці таблиці). В другої клітинці таблиці розташовано стаціонарний текст. Задавання стилю тега <MARQUEE> дозволяє придати йому більшу виразність.

Програмний код рядка, що біжить вгору:

```
<HTML>
<HEAD>
<STYLE type="text/css">
  MARQUEE (background-color: lime;
      border: fuchsia solid;
      height: 200pt; width: 72pt;
      font-size: 16pt; color: red)
  P (color: navy; font-size; 30pt; text-align: left)
</STYLE>
</HEAD>
```

```
<BODY>
<TABLE>
  <TR>
      <TD>
        <MARQUEE direction=”up”>
      Леонід Мараховський  і Олег Лемешев
        </MARQUEE>
      </TD>
      <TD>
      <P>Комп’ютерні </P>
      <P>мережі</P>
      <P>і телекомунікація </P>
      </TD>
   </TR>
</TABLE>
</BODY>
</HTML>
```

## 14.8. Плаваючий фрейм

Плаваючий фрейм — це область, яку можна завантажити на Web-сторінку. Плаваючий фрейм створюється за допомогою парного тега <IFRAME>. Атрибут SRC задає **URL**-адресу документа, який відображається у плаваючому вікні. Синтаксис:

```
<IFRAME
      ALIGN=ABSBOTTOM | ABSMIDDLE | BASELINE | BOTTOM | LEFT |
      MIDDLE | RIGHT | TEXTTOP | TOP
      BORDER=pixels
      CLASS=classname
      DATAFLD=colname
      DATASRC=#ID
      FRAMEBORDER=NO | YES | 0 | 1
```

FRAMESPACING=*pixels*

HEIGHT=*n*

HSPACE=*pixels*

ID=*value*

LANG=*language*

LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS

MARGINHEIGHT=*pixels*

MARGINWIDTH=*pixels*

NAME=*window_name* | _blank | _parent | _self | _top

NORESIZE=NORESIZE | RESIZE

SCROLLING=AUTO | NO | YES

SRC=*url*

STYLE=*cssl-properties*

TITLE=*text*

VSPACE=*pixels*

WIDTH=*n*

>

В наступному прикладі при уведенні в поле уведення **URL-**адреси документа і натиснення кнопки **Show if** в плаваючому вікні відображається вибраний документ.

Програмний код плаваючого фрейма:

```
<HTML>
<HEAD>
<STYLE>
<!--
      .CMD {color: navy; fond-size: 110%;}
-->
</STYLE>
<SCRIPT language = “VBScript” >
<!--
Sub cmdImage_OnClick()
      ifrShow.location.href = txtImage.value
End Sub
```

```
-->
</SCRIPT>
</HEAD>
<BODY>
<INPUT id="txtImage" value="Filter2.htm" >
<INPUT type="button" id="cmdImage" value="Show it" class="cmd">
<BR><HR><BR>
<P align="center">
<IFRAME id="ifrShow" frameborder="yes" scrolling="Yes"
      heiht=200 width=200 align="bottom" src="Filter2.htm">
</IFRAME>
</BODY>
</HTML>
```

## 14. 9. Питання для самоконтролю

1. Як старіє інформація?
2. Чим нас приваблює можливості Web-документів?
3. Які програмні засоби дозволяють динамічний обмін між користувачем і сервером?
4. Для чого застосовується форма в HTML?
5. Які теги можна розташовувати між парами тегів <FORM>…</FORM> та що вони задають?
6. Яким тегом створюються елементи управління на формі?
7. Яка різниця в назві події HTML і VBScript та в VBA?
8. Коли генерується подія onBlur елементом управління?
9. За допомогою значення якого атрибуту задається спливаюча підказка?
10. За допомогою обробки яких подій можна при натиснутій кнопці змінити її колір та напис на поверхні кнопки ?
11. За допомогою якого тега можна розфарбувати кнопки як на їх поверхні, так і слова на них?

12. Яка властивість поля уведення надає або встановлює значення, що відображається в полі уведення?

13. Яка властивість прапорця приймає значення True, коли прапорець встановлено, і False, коли прапорець скинуто?

14. Що можна використати для створення масиву елементів в Visual Basic?

15. Яка властивість повертає активний елемент сторінки?

16. Яку властивість застосовують для ідентифікації активного елемента сторінки?

17. Як досягається візуальне групування елементів управління, в результаті чого ці елементи управління будуть оточені рамкою?

18. Якими парними тегами створюється заголовок рамки?

19. Які властивості visibility об'єкту style застосовують, щоб бачити або сховати елемент?

20. Які властивості display об'єкту style застосовують, щоб бачити або сховати елемент?

21. Яка різниця при застосуванні властивостей visibility і display?

22. За допомогою якої обробки події дані, що вводяться користувачем, можна перевіряти на етапі їх набору?

23. Який парний тег в багатьох броузера подається як меню або список вибору?

24. Який подвійний тег створює багаторядкове поле введення?

25. Якими тегами створюються картки-зображення (imagemapes)?

26. Який єдиний атрибут має парний тег <MAP>?

27. Який атрибут здійснює зв'язок між карткою і конкретним зображенням, що створено за допомогою тега <IMG>?

28. Який тег застосовується для створення біжучого рядка в HTML?

29. Що таке плаваючий фрейм? Що задає атрибут SRC?

30. За допомогою якого парного тега створюється плаваючий фрейм?

## 15. ВИКОРИСТАННЯ ЕЛЕМЕНТІВ КЕРУВАННЯ

Парний тег<OBJECT> дає можливість завантажити в HTML-документи зображення, апплети, документи і елементи ActiveX. Синтаксис:

<OBJECT

ACCESSKEY=*key*

ALIGN=ABSBOTTOM | ABSMIDDLE | BASELINE | BOTTOM | LEFT | MIDDLE | RIGHT | TEXTTOP | TOP

CLASS=*classname*

CLASSID=*id*

CODE=*url*

CODEBASE=*url*

CODETYPE=*media-type*

DATA=*url*

DATAFLD=*colname*

DATASRC=#*ID*

HEIGHT=*n*

ID=*value*

LANG=*language*

LANGUAGE=JAVASCRIPT | JSCRIPT | VBSCRIPT | VBS

NAME=*name*

STYLE=*cssl-properties*

TABINDEX=*n*

TITLE=*text*

TYPE=*MIME-type*

WIDTH=*n*

*event = script*

>

□ CLASSID — ідентифікатор об’єкта.

□ CODEBASE — URL, що визначає розташування об’єкта. Дозволяє завантажити компонент на локальний комп’ютер по мережі.

□ CODETYPE — визначає тип кодування об’єкта.

Властивості об’єкта задаються за допомогою тегів <PARAM>, які розташовуються всередині тега-контейнера <OBJECT>. Синтаксис:

```
<PARAM
    DATAFLD=colname
    DATAFORMATAS=HTML | TEXT
    DATASRC=#ID
    NAME=name
    VALUE=value
>
```

□ NAME — ім’я властивості.

□ VALUE — значення властивості.

*Елемент управління Structured Graphics*

ActiveX-елемент управління **Structured Graphics** (або як скорочення **SG**) дозволяє створювати складні графічні ефекти на Web-сторінці.

Елемент управління **SG** створюється в коді інструкцією:

```
<OBJECT ID=object STYLE="WIDTH:width; HEIGHT: height; Z-INDEX:
    z-index" CLASSID="CLSID:369303C2-D7AC-11d0-89D5-
    00A0C90833E6">
  <PARAM NAME="PropertyName" VALUE="Value">
</OBJECT>
```

Перерахуємо можливі значення параметра *PropertyName.*

| **Властивість** | **Опис** |
|---|---|
| CoordinateSystem | Коли значення дорівнює 0, то вісь ординат прямує вниз, а коли 1, то вгору |
| DrawingSurface | Повертає або встановлює поверхню Microsoft DirectAnimation, на якій створюється малюнок |
| ExtentHeight, ExtentWidth, ExtentLeft, ExtentTop | Встановлюють висоту, довжину і координати верхнього лівого кута об’єкта в пікселях |
| Library | Задає бібліотеку DirectAnimation Library |

| | |
|---|---|
| MouseEventsEnabled | Визначає, чи реагує об'єкт на події, які генеруються мишею |
| PreserveAspectRatio | Задає, чи може малюнок масштабуватися пропорційно |
| SourceURL | Дозволяє використовувати зовнішній файл в якості структурної графіки |

В тег-контейнері <OBJECT> елемента управління **SG** можна встановлювати не тільки властивості, але і застосовувати методи. Синтаксис:

```
<OBJECT ID=object STYLE="WIDTH:width; HEIGHT: height; Z-INDEX:
    z-index" CLASSID="CLSID:369303C2-D7AC-11d0-89D5-
    00A0C90833E6">
  <PARAM NAME="LINEnnnn" VALUE="method">
</OBJECT>
```

Перерахуємо методи елемента управління **SG,** які використовуються як в тезі <PARAM>, так і застосовуються до об'єкту **SG** в коді.

| **Метод** | **Опис** |
|---|---|
| SetFillColor | Задає колір ліній або фігур |
| SetFillStyle | Задає тип заповнення |
| SetFont | Задає шрифт разом з методом Text |
| SetGradientFill | Задає прямокутну область, яка заповнюється градієнтом. Цей метод використовується тільки при значенні параметра type метода SetFillStyle, що дорівнює 11 |
| SetHatchFill | Встановлює, чи прозора штрихова |
| SetLineColor | Задає колір лінії в RGB-форматі |
| SetLineStyle | Задає тип лінії |
| SetGradientShape | Задає форму градієнта |
| SetTextureFill | Задає графічний файл, який використовується для створення фону на фігурі |
| Text | Задає текст |
| Arc, Pie | Створює еліптичну дугу і сегмент еліпса |

| | |
|---|---|
| FillSpline | Створює сплайн на основі якоїсь послідовності крапок |
| Oval | Створює еліпс |
| Polygon,PolyLine | Створює замкнутий багатокутник і ламану лінію |
| PolySpline | Створює ламану лінію, ланками якої є сплайни |
| Rect | Створює прямокутник |
| RoundRect | Створює прямокутник з закругленнями кутів |

Перерахуємо також методи елемента управління **SG**, які застосовуються до нього тільки в коді.

| **Метод** | **Опис** |
|---|---|
| Clear | Вилучає графічні зображення. Використовується перед тим, як почати створювати зображення за допомогою властивості DrawingSurface |
| Rotate | Повертає об’єкт навколо x-, y- і z-вісей |
| Scale | Вводить новий масштаб по вісях |
| SetIdentity | Задає початкові установки об’єкта |
| Translate | Встановлює нові початкові координати для об’єкта |

Розглянемо приклад використання елемента управління **Structured Graphics** для створення анімації.

```
<HTML>
<BODY bgcolor=navy>
<OBJECT classid=clsid:369303C2--D7AC-11d0-89D5-
      00A0C90833E6 id=objSG  style=”HEIGHT: 400px; POSITION:
      absolute; TOP: 10px; WIDTH: 400px; LEFT; 10; Z-INDEX: -1”>
  <PARM name=”SourceURL” value=””>
  <PARM name=”CoordinateSystem” value=”0”>

  <PARM name=”MouseEventsEnabled” value=”0”>
  <PARM name=”PreserveAspectRatio” value=”1”>
</OBJECT>
```

```
<SCRIPT language=”VBScript”>
dim ds
Set lib=objSG.Library
Set ds= objSG.DrawSurface
ds.fillcolor lib.green
ds.rect -50, -50, 200, 200
ds. fillcolor lib.ColorRgb255(0,200,255)
ds.oval -30, -30, 90, 90
ds.SaveGraphicsState ()
ds.transform lib.rotate2rate (5)
ds. fillcolor lib.ColorRgb255(255,100,0)
ds.rect -10, -100, 70, 200
ds.RestoreGraphicsState ()
objSG.DrawSurface=ds
RotateIt
Sub RotateIt ()
      objSG.rotate 2,0,2
      window.setTimeout “call RotateIt”, 50
End Sub
</SCRIPT>
</HTML>
```

## 16. ЗВ’ЯЗУВАННЯ WEB-СТОРІНКИ З БАЗОЮ ДАНИХ

ActiveX-елемент управління **Tabular Data Control** (або скорочено TDC) дозволяє зв’язувати дані, що зберігаються в текстовому файлі, з Web-сторінкою.

Створимо Web-сторінку, на якій маємо два поля уведення: на першому буде відображатися прізвище, а на другому — оцінка студента. Крім того, на Web-сторінці розташуємо чотири кнопки, які дозволять перейти на перший, попередний, наступний і останній запис бази даних.

Інформація про студентів зберігається в базі даних, в текстовому файлі (наприклад, у файлі dtResult.txt), який треба створити до того, як почнемо будувати Web-сторінку. В нашому випадку файл dtResult.txt має таку структуру.

*Лістінг. Перегляд записів в базі даних. Файл dtResult.txt*

Прізвище, Оцінка

Махно, 4

Кузько, 3

Скопень, 3

Будя, 5

Квакова, 4

Зверніть увагу, що в першому рядку файла розміщені назви полів, а роздільниками між ними є кома.

Елемент управління TDC створюється в коді інструкцією

```
<OBJECT classid="clsid: 333C7ИС4-460A-11B0-ИС04-0080C7055Ф83">
  <PARM name="PropertyName" VALUE="Value">
</OBJECT>
```

Перерахуємо можливі значення параметра *PropertyName*.

| **Властивість** | **Опис** |
|---|---|
| ChartSet | Визначає таблицю символів, яка використовується для подання даних |
| DataURL | URL файла даних |
| EscapeChar | Визначає символ, який використовується в таблиці в якості Escape-символа |
| FieldDelim | Визначає символ, який використовується в таблиці в якості роздільника. За замовчуванням застосовується кома |
| Language | Визначає мову подання даних. За замовчуванням — eng-us |
| TextQualifier | Визначає символ, який використовується для оточення поля |
| RowDelim | Визначає символ, який використовується для ідентифікації кінця запису |
| UseHeader | Визначає, чи являється перший рядок файла даних заголовками полів таблиці |

Для прив'язування елемента документа до бази даних застосовують два атрибути.

| Атрибут | Опис |
|---|---|
| DATASRC | Вказує об’єкт джерела даних, який постачає дані |
| DATAFLD | Вказує іменований стовпчик або поле, яке повинно бути зв’язане з об’єктом джерела даних |

Програмний код завершення створення Web-сторінки.

```
<HTML>
<HEAD>
<STYLE>
<!--
TD{font-family: Arial, san-serif; font-weight:bold;
      font-size: 110%;background-color:agua; color: red}
.CMD{font-family: Arial, sans-serif; font-weight:bold;
      font-size: 90%;color: navy;}
.TXT{font-family: Arial, sans-serif; font-weight:bold;
      font-size: 70%}
-->
<SCRIPT language = “VBScript” >
<!--
Sub cmdFirst_OnClick ()
      Document.frmGrade.datGrade.Recordset.MoveNext
      If Document.frmGrade.datGrade.Recordset.EOR Then
      Document.frmGrade.datGrade.Recordset..MoveLast
      End If
End Sub
Sub cmdPrev_OnClick
      Document.frmGrade.datGrade.Recordset.MovePrevious
      If Document.frmGrade.datGrade.Recordset.BOF Then
        Document.frmGrade.datGrade.Recordset.MoveFirst
      End If
End Sub
Sub cmdLast_OnClick ()
```

```
        Document.frmGrade.datGrade.Recordset.MoveLast
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FORM name="frmGrade">
<OBJECT  classid="clsid:333C7BC4-460F-11D0-DC04-0080C7055A83"
        id=datGrade>
   <PARAM name="UseHeader" value="1">
   <PARAM name="DataURL" value="dtResult.txt">
</OBJECT>
<TABLE>
   <TR>
        <TD>Прізвище:</TD>
        <TD><INPUT class="txt" id="txtGrade" type="TEXT" size=3
        datasrc=#datGrade datafld="Оцінка"></TD>
    </TR>
</TABLE>
<INPUT class="cmd" type="button" name="" value="<<">
<INPUT class="cmd" type="button" name="" value=" < ">
<INPUT class="cmd" type="button" name="" value=" > ">
<INPUT class="cmd" type="button" name="" value=">>">
</FROM>
</BODY>
</HTML>
```

## 16.1. Перегляд бази даних з графічним зображеннями

Як базу даних можна використовувати також малюнки. Для цього в текстовому файлі БД треба вказати посилання на відповідний графічний файл, а зображення виводити в тег <IMG> шляхом зв'язування його з полем БД через атрибут DATAFLD.

Розглянемо це на прикладі текстової БД dtAssistant.txt, в якій містяться імена помічників MS Office, їх коротенький опис і зображення.

*Лістінг*. Помічники MS Office. Файл dtAssistant.txt

```
Имя; Описание; Рисунок
Бобик; Свалившись в ущелье, зовите Лесси. Но если вам нужна помощь в офисе, зовите Бобика.; Dog.gif
Мурка; Если вы охотитесь за вопросами на территории Microsoft Office, то Мурка поможет вам их выследить.; Cat.gif
Ученый; Мозг ученого работает со скоростью света. Воспользуйтесь им для того, чтобы сэкономить ваши усилия.; Sci.gif
```

*Лістінг.*Помічники Microsoft Office. Web-сторінка

```
<HTML>
<HEAD>
<STYLE>
  I { color: green; font-weight: bold; text-decoration: underline}
  BUTTON {cursor: hand}
</STYLE>
<SCRIPT language = “VBScript” >
<!--
Sub moveIt (Go_To)
  Select Case Go_To
      Case “Up”
        If objData.recordset.AbsolutePosition _
            < objData.recordset.RecordCount Then
              objData.recordset.MoveNext
        Else
            alert “Останій запис”
        End If
      Case “Down”
        If objData.recordset.AbsolutePosition > 1 Then
              objData.recordset.MovePrevious
        Else
            alert “Перший запис”
```

```
        End If
      End Select
End Sub
-->
</SCRIPT>
</HEAD>
<BODY bgcolor="yellow">
<FONT size=3>
<H2>Помічники Microsoft Office</H2>
<OBJECT id="objData" width="0" height="0"
       classid="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83">
  <param name="dATAurl" value="dtAssistant.txt">
  <param name="fIELDdELIM" value=";">
  <param name="UseHeader" value="True">
</OBJECT>
<P>
<I>Ім'я:</I><SPAN datasrc=#objData datafld="Ім'я"></SPAN>
</P>
<P>
<I>Опис:</I><SPAN datasrc=#objData datafld="Опис"></SPAN>
</P>
<P>
<I>Малюнок:</I><SPAN datasrc=#objData datafld="Малюнок"></SPAN>
<HR>
<BUTTON onclick="moveIt ('Down')>Назад</BUTTON>
<BUTTON onclick="moveIt ('Up')>Вперед</BUTTON>
</BODY>
</HTML>
```

## 16.2. Створення таблиць на Web-сторінці на основі бази даних

Тег <TABLE> — це контейнер для зв'язків з даними, які безпосередньо не встановлюються. Розглянемо це на прикладі бази даних з результатами екзаменів, що зберігаються в текстовому файлі dtResult.txt. Зміст БД виведемо

в таблицю, створювану на Web-сторінці. При цьому зробимо так, щоб при натисненні на заголовку стовпчика **Прізвище** дані сортувалися у порядку зростання спочатку по стовпчику **Прізвище**, а у випадку співпадання прізвищ, — по стовпчику **Оцінка**. При натисненні на заголовку стовпчика **Оцінка** дані упорядковуються по цьому стовпчику за зменшенням.

Тег <TABLE> зв'язується з БД за допомогою встановлення значення атрибута DATASRC, рівним URL текстового файла, з БД. Прив'язка окремих полів даних до стовпчиків таблиці виконується за допомогою атрибута DATAFLD тега <DIV>. При цьому зв'язування кожного рядка таблиці з записом БД виконується автоматично.

Для сортування і фільтрації даних треба використовувати властивості Sort і Filter елемента управління TDC. Метод Reset () виконує сортування або фільтрацію даних, що задають властивості Sort і Filter, і відображення результата у вікні броузера.

*Літинг*. Програмний код створення таблиць.

```
<HTML>
<HEAD>
<STYLE>
<!--
  .Theader{color: navy; font-weight: bold;
      text-decoration: underline; cursor: hand}
-->
</STYLE>
<SCRIPT language = "VBScript" >
<!--
Sub tdSurname_OnClick ()
      dbGrade.Sort = "Прізвище;Оцінка"
      dbGrade.Reset ()
End Sub
Sub tdGrade_OnClick ()
      dbGrade.Sort = "–Оцінка"
      dbGrade.Reset ()
```

```
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FONT face="verdana,arial" size=2>
<H2>Результати іспиту</H2>
<OBJECT  classid="clsid:333C7BC4-460F-11D0-BC04-0080C7055A83"
      id="dbGrade">
  <PARAM name="DataURL" value="dtResult.txt">
  <PARAM name="UseHeader" value="True">
</OBJECT>
<TABLE id=tbGrade datasrc=#dbGrade border=1>
<THEAD>
  <TR>
    <TD class="Theader"><DIV id="tdSurname">Прізвище</DIV></TD>
    <TD class="Theader"><DIV id="tdGrade">Оцінка</DIV></TD>
  </TR>
</THEAD>
<TBODY>
  <TR>
    <TD ><DIV datafld="Прізвище"></DIV></TD>
    <TD><DIV datafld="Оцінка"></DIV></TD>
  </TR>
</TBODY>
</TABLE>
</BODY>
</HTML>
```

### 16.3. Фільтрування даних

Розглянемо як приклад текстовий файл dtResult.txt з оцінками і прізвищами студентів. Створимо на Web-сторінці два списка, в першому з яких виводиться найменша, а в другому — найбільша оцінка діапазону. Після вибору значень в

цих двох списках в таблиці на Web-сторінці буде відображатися інформація тільки про тих студентів, оцінки яких знаходяться у вибраному діапазоні.

*Лістінг*. Програмний код фільтрації даних.

```
<HTML>
<HEAD>
<STYLE>
<!--
  .Theader{color: navy; font-weight: bold;}
-->
</STYLE>
<SCRIPT language = “VBScript” >
<!--
Sub window_onLoad ()
      lstGradeFrom.SelectedIndex=3
      lstGradeTo.SelectedIndex=0
End Sub
Sub ShowResult ()
      Dim Cond
      Cond=”Оцінка>=” & Cstr(5-lstGradeFrom.SelectedIndex) & _
            “&” & “Оцінка<=” & Cstr(5-lstGradeTo.SelectedIndex)
      dbGrade.Filter =Cond
      dbGrade.Reset ()
End Sub
-->
</SCRIPT>
</HEAD>
<BODY>
<FONT face=”verdana,arial,helvetica” size=2>
<H2>Результати іспиту</H2>
<OBJECT  classid=”clsid:333C7BC4-460F-11D0-BC04-0080C7055A83”
      id=”dbGrade”>
  <PARAM name=”DataURL” value=”dtResult.txt”>
  <PARAM name=”UseHeader” value=”True”>
```

```
</OBJECT>
<TABLE border=1>
  <TR>
    <TD >Від</TD>
    <TD>До</TD>
  </TR>
  <TR>
    <TD>
      <SELECT name=”lstGradeFrom” OnChange=”ShowResult ()>
            <OPTION name=”opt5”>5
            <OPTION name=”opt4”>4
            <OPTION name=”opt3”>3
            <OPTION name=”opt2”>2
</TD>
    <TD>
      <SELECT name=”lstGradeTo” OnChange=”ShowResult ()>
            <OPTION name=”opt5”>5
            <OPTION name=”opt4”>4
            <OPTION name=”opt3”>3
            <OPTION name=”opt2”>2
  </TD>
 </TR>
</TABLE>
<TABLE id=”tblgrade” datasrc=#dbGrate border=1>
  <THEAD>
  <TR>
  <TD CLASS=”Theader”><DIV id=”tdSurname”>Прізвище</DIV></TD>
  <TD CLASS=”Theader”><DIV id=”tdSurname”>Оцінка</DIV></TD>
  </TR>
</THEAD>
<TBODY>
  <TR>
    <TD ><DIV datafld=”Прізвище”></DIV></TD>
```

```
    <TD><DIV datafld="Оцінка"></DIV></TD>
  </TR>
</TBODY>
</TABLE>
</BODY>
</HTML>
```

## 16.4. Питання для самоконтролю

1. Який парний тег дає можливість завантажити в HTML-документи зображення, апплети, документи і елементи ActiveX?
2. Який ActiveX-елемент управління дозволяє створювати складні графічні ефекти на Web-сторінці?
3. Як створюється елемент управління Structured Graphics?
4. Перерахуйте можливі значення параметра PropertyName.
5. Чи можна застосовувати в тег-контейнері <OBJECT> елемента управління Structured Graphics властивості і методи?
6. Перерахуйте методи елемента управління Structured Graphics, які використовуються в тезі <PARAM>?
7. Перерахуйте методи елемента управління Structured Graphics, які застосовуються до нього тільки в коді?
8. Чи дозволяє ActiveX-елемент управління Tabular Data Control зв'язувати дані, що зберігаються в текстовому файлі, з Web-сторінкою,?
9. Які два атрибути застосовуються до бази даних для прив'язування елемента документа?
10. Чи можна використовувати малюнок як базу даних?
11. Що треба зробити, щоб можна було використовувати малюнки як базу даних?
12. Яке призначення тега <TABLE>?
13. За допомогою якого атрибута тег <TABLE> зв'язується з БД?
14. За допомогою якого тега і його атрибута виконується прив'язка окремих полів даних до стовпчиків таблиці?

15. Які властивості елемента управління TDC треба використовувати для сортування і фільтрації даних?

## 17. МОДАЛЬНЕ ВІКНО

В HTML-документі можна створювати модальні вікна для перегляду. Таке вікно буде активним до тих пір, поки користувач його не закриє. Це вікно застосовуються для створення користувацьких вікон виведення інформації або вікон **About**. Модальне вікно відкривається за методом showModalDialog об'єкта window. Синтаксис:

showModalDialog (*sURL* [, *vArguments* [, *sFeatures*]])

□ *sURL* — URL документа, завантаженого в модальне вікно.

□ *vArguments* — передає в діалогове вікно довільний набір параметрів. Безпосереднє передавання даних виконується за допомогою метода dialogArguments об'єкта window. Повернення ж значень із діалогового вікна в документ реалізується властивістю returnValue того ж об'єкта

□ *sFeatures* — параметри, що задають зовнішній вигляд вікна. Допустимі значення:

- dialogWidth:*number* — довжина вікна;
- dialogHeight:*number* — висота вікна;
- dialogTop:*number* — ордината верхнього лівого кута вікна;
- dialogLeft:*number* — абсциса верхнього лівого кута вікна;
- center: {yes | no | 1 | 0} — розташування вікна по центру екрана.

### 17.1. Вікно About

Розглянемо як за допомогою модального вікна конструюється вікно **About**, що містить інформацію про документ (наприклад, відомості про автора, час створення тощо). Зміст вікна **About** створюється в окремому файлі About.htm, який треба розташувати у тому ж каталозі, де є основний файл. В основному файлі створюється документ, в якому розташована кнопка **About**. Натиснення на цю кнопку відкриває модальне вікно **About.**

*Лістінг*. Основний документ з кнопкою виклика модального вікна **About**

```
<HTML>
<SCRIPT language = “VBScript” >
```

```
<!--
Sub cmdAbout_onClick ()
      window.showModalDialog “About.htm”, “”, _
                  “dialogHeight:15;dialogTop:0;dialogLeft:0”
End Sub
-->
</SCRIPT>
<BODY>
      <BUTTON id=”cmdAbout”>About</BUTTON>
</BODY>
</HTML>
```

*Лістінг.* Модальне вікно About. Код із файла About.htm

```
<HTML>
<HEAD>
      <TITLE>Про програму</TITLE>
</HEAD>
<BODY style=”background:lightgrey”>
  <CENTER>
      <H2>Приклад модального діалогового вікна</H2>
      <H3>Андрій Михно</H3>
      <SPAN style=”font-style: italic; color: navy”>
        Найпростіший засіб конструювання діалогового вікна!
      </SPAN>
      <P>
      <INPUT type=”button” value=” OK ” onclick=”window.close ()>
  </CENTER>
</BODY>
</HTML>
```

## 17.2. Обмін даними між вікнами

Користувацьке вікно уведення даних створюється файлом Input.htm, який треба розташовувати в тому ж каталозі, що і основний файл. В початковому документі маємо поле уведення **Name**, кнопка **Input** і багаторядкове поле

уведення, що недоступне для користувача. Натиснення кнопки **Input** приводить до того, що на екрані відображається користувацьке вікно уведення з трьома полями: **Name, Email** і **Message** і кнопкою **OK**, причому в полі уведення **Name** виводиться ім'я, яке було вказано в однеіменному полі основного документа. Після уведення даних і натиснення кнопки **OK** користувацьке вікно уведення даних закривається, а уведенні дані відображаються в багаторядкове поле уведення документа.

*Лістінг*. Виклик вікна уведення даних

```
<HTML>
<SCRIPT language = "VBScript" >
<!--
Sub cmdInput_onClick ()
      Dim vReceive, vSend
      vSend = txtName.value
      vReceive=window.showModalDialog ("Input.htm", vSend, _
      "dialogHeight:20")
      txtReceive.value = ""
      txtReceive.value = vReceive
End Sub
-->
</SCRIPT>
<BODY>
Name <INPUT id="txtName" type="textbox">
<BUTTON id="cmdInput">Input</BUTTON>
<P>
<TEXTAREA id="vReceive" disabled></TEXTAREA>
</BODY>
</HTML>
```

*Лістінг*.Модальне вікно уведення даних.Код файла Input.htm

```
<HTML>
<HEAD>
<TITLE>User Inputbox</TITLE>
```

```
<SCRIPT language = “VBScript”>
<!--
Sub page_Initialize ()
      frmInput.txtName.bvalue = window.dialogArguments
End Sub
Sub SendItBack ()
      Dim vReturn
      vReturn=frmInput.txtEMail.value
      vReturn=vReturn & chr (13) & frmInput.txtMessage.value
      window.returnValue = vReturn
      window.event.returnValue =false
      window.close
End Sub
-->
</SCRIPT>
</HEAD>
<BODY style=”background: lightgrey; align: left”
      OnLoad=”Page_Initialize ()>
<FROM id=”frmInput”>
      Name <INPUT id=”txtName”>
      <P>
      Email <INPUT id=”txtEmail”>
      <P>
      Message <TEXTAREA id=”txtMessage”>
      </TEXTAREA>
      <P>
      <INPUT type=”button” Value=”  OK  ” onClick=”SendItBack ()>
</FROM>
</BODY>
</HTML>
```

## 17. 3. Питання для самоконтролю

1. Чи можна в HTML-документі створювати модальні вікна для перегляду?
2. До яких пір модальне вікно буде активним?
3. Для чого застосовуються модальні вікна?
4. За яким методом відкриваються модальні вікна?
5. За допомогою якого методу виконується безпосереднє передавання даних?
6. Як реалізується повернення значень із діалогового вікна в документ?
7. Які параметри задають зовнішний вигляд вікна?
8. Що відображає значення dialogWidth:*number?*
9. Що відображає значення dialogHeight:*number?*
10. Що відображає значення dialogTop:*number?*
11. Що відображає значення dialogLeft:*number?*
12. Що відображає значення center: {yes | no | 1 | 0}?
13. В якому файлі створюється зміст вікна About?
14. В якому файлі створюється документ, в якому розташована кнопка About?
15. Яким файлом створюється користувацьке вікно уведення даних, і який треба розташовувати в тому ж каталозі, що і основний файл

# 18. ЕЛЕМЕНТИ КЕРУВАННЯ ACTIVEX

## 18. 1. Перший елемент керування ActiveX

Створимо за допомогою Visual Basic 6.0 елемент керування **ActiveX**, в якому відображається миготливий напис.

Зробимо запуск Visual Basic 6.0. У вікні Visual Basic виберемо значок **ActiveX Control** і натиснемо кнопку **ОК**. З'явиться середовище розробки елемента керування **ActiveX** (рис.12.3.).

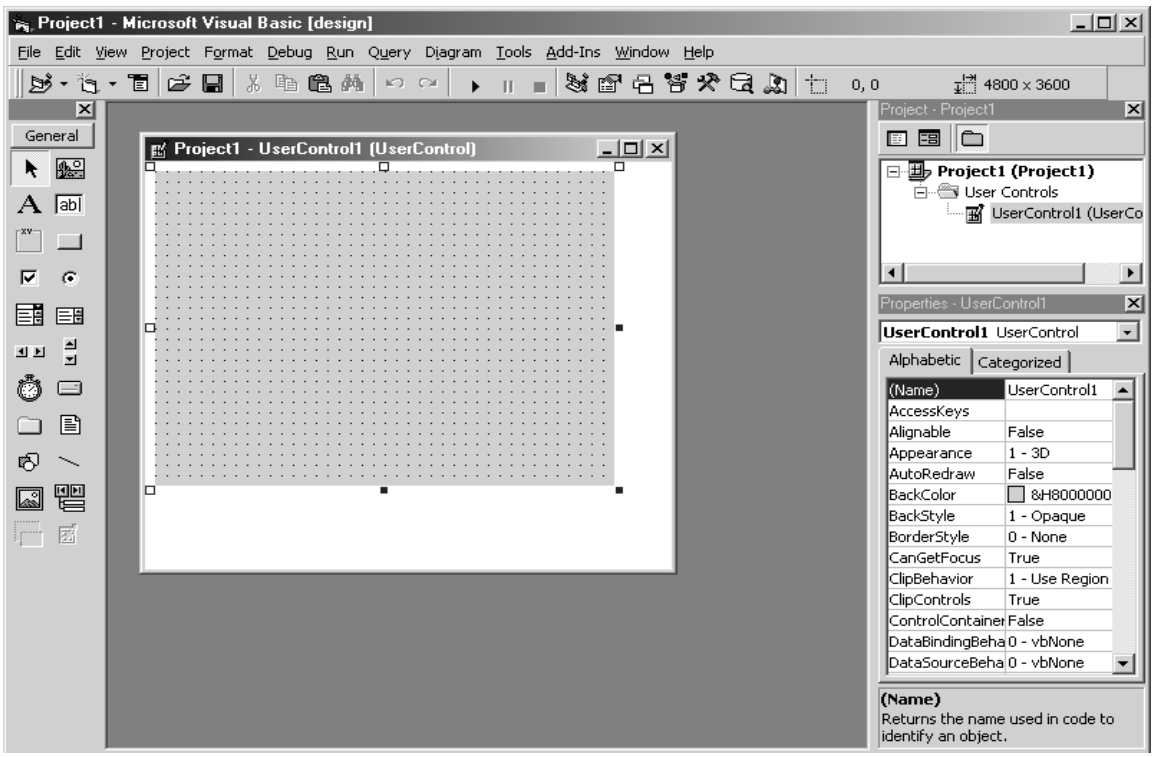



**Рис.12. 3.** Середовище розробки елемента керування **ActiveX**

На початку розробки проекту виберемо команду **Project | Properties** і в вікні **Project Properties** в поле **Project | Properties Name** введемо ім’я проекту — Blinker.

Натискаємо кнопку **ОК** (рис.12.4.). В вікні **Properties** встановвлюємо для об’єкта **UserControl** значення властивості Name рівним ctlBlinker (рис. 12.5.).

Створимо проект. Розташуємо в вікні **UserControl** елемент управління **Timer** (рис. 12.6.), а в модулі **UserControl** наберемо такий програмний код.

*Лістінг*. Перший елемент керування ActiveX. Миготливий напис

```
Const sText = "Andrey Michno"
'Рядок зображуваного тексту
```

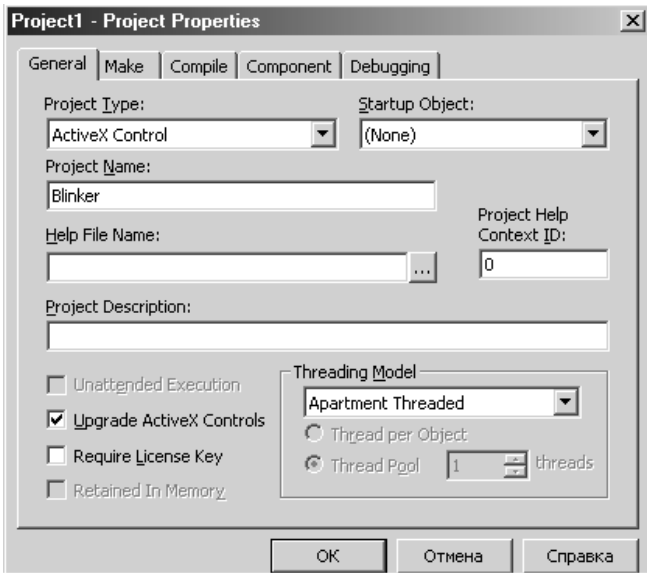



**Рис. 12. 4**. Вікно **Project Properties**

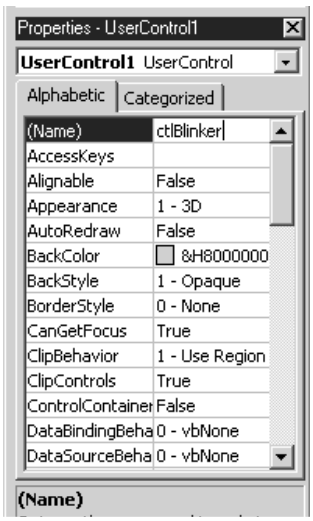



**Рис. 12. 5**. Вікно **Properties**

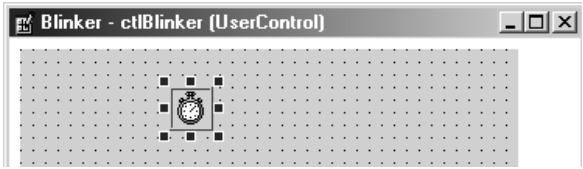



**Рис. 12. 6.** Елемент керування Timer

```
Private Sub UserControl_Initialize()
    Timer1.Interval = 100
    With UserControl
        .Font.Bold = True
        .Font.Size = 12
        .BackColor = vbBlack
    End With
End Sub
Private Sub UserControl_Terminate()
    Timer1.Enabled = True
End Sub

Private Sub UserControl_Resize()
'Довжина і висота елемента управління рівні
'довжина і висота відображаємого тексту, котрі повертаються
'функціями TextWidth і TextHeight
    UserControl.Width = TextWidth(sText)
    UserControl.Height = TextHeight(sText)
'Активація елемента управління (мигатінія)
'здійснюється тільки на етапе виконання проекту,
'а не його конструювання
    If UserControl.Ambient.UserMode Then
        Timer1.Enabled = True
     Else
        Timer1.Enabled = False
    End If
End Sub

Private Sub Timer1_Timer()
    Static bCount As Boolean
    If Ambient.UserMode Then
'Зміний вивід напису жовтим і чорним кольором
```

```
    Select Case bCount
       Case True
          UserControl.ForeColor = vbYellow
       Case False
           UserControl.ForeColor = vbBlack
    End Select
    UserControl.CurrentX = 0
    UserControl.CurrentY = 0
    Print sText
    bCount = Not bCount
End If
End Sub
```

Робота цього проекту здійснюється процедурою Timer1_Timer, яка виводить текст, змінюючи його колір. Головною задачею таймера є запуск вказаних дій в процедурі обробки події Timer цього елемента керування через визначені проміжки часу, які задані значенями властивості Interval в мілісекундах. Іншою важливою властивістю таймера є Enabled. Для функціонування таймера значення властивості Enabled повинно бути встановлено рівним True. При значенні False таймер відключений. Елемент управління **Timer** ніколи не відображається в вікні в час виконання програми. Таким чином, його можна розташовувати де завгодно на формі.

Для завершення роботи з проектом треба його зберегти і створити осх-файл. Для збереження проекту треба виконати команди **File | Save Project As.** На екрані з'явиться вікно **Save File As**, яке запропонує вам спочатку зберегти ctl-файл (файл ActiveX-елемента з його описом і кодом). В поле **Имя файла** введіть Blinker і натисніть кнопку **Сохранить.** Потім з'явиться вікно, в якому пропонується зберегти vbp-файл (файл проекту). В поле **Имя файла** введіть Blinker і натисніть кнопку **Сохранить.** Таким чином, на диску з'являться два файли Blinker.ctl і Blinker.vbp. Після збереження двох файлів треба відкомпілювати проект. Для цього виконаємо команду **File | Make**

**ctlBlinker.ocx**. На екрані з’явиться вікно **Make Project**. В поле **Имя файла** введіть ctlBlinker і натисніть кнопку **ОК**. Проект відкомпілюваний, а на диску з’явиться ще один файл ctlBlinker.ocx.

Перевірку роботи можна зробити двома способами:

- виконати команду **Run | Start** або натиснути кнопку [F5] і в вікні Internet Explorer побачити його роботу;
- запустити MS Excel. Виконати команду **Сервис | Макрос |** Редактор **Visual Basic**. В редакторі Visual Basic в проект додати форму. Виконати каманду **Tools | Additional Controls.** В вікні **Дополнительные элементы** у списку **Доступные элементы управления** вибрати прапорець **Blinker.ctlBlinker**. Натиснути кнопку **ОК**. В панелі інструментів буде додана нова кнопка **ctlBlinker** (рис. 12.7.). Треба перемістити екземпляр елемента управління ctlBlinker на форму. Запустити програму клавішею [F5], щоб побачити його роботу в текстовому проекті.

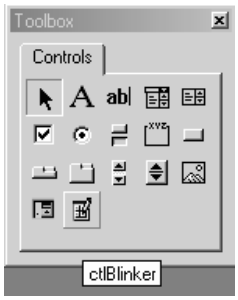



**Рис. 2. 8**. Кнопка ctlBlinker

### *16. 2. Реєстрація елементів керування ActiveX*

Для використання на ПЕОМ елемент керування **ActiveX** треба його заєреструвати (занести його в реєстр). Програма VB виконує автоматично реєстрацію при компіляції конструйованих елементів керування **ActiveX**. Однак при їх розповсюджені також треба забезпечити таку реєстрацію.

Програма VB Package and Deployment Wizard (Майстер упаковки і розповсюдження) дозволяє створити із побудованого проекту інсталяційний пакет з програмою установки Setup.exe, при запуску якої здійснюється автоматична реєстрація на ПЕОМ користувача. Майстер упаковки і розповсюдження додається в середовище розробки VB командою **Add-Ins | Add-In Manager**.

Реєстрацію можна провести також за допомогою файла Regsvr32.exe (розташованого разом з VB), виконуючи DOS-команду Regsvr32.exe , де параметром вказується ім’я ocx-файла, що реєструється. Наприклад,

Regsvr32.exe ctlBlinker.ocx

Для вилучення із реєстра зареєстрованого ocx-файла, треба виконати команду з опцією /U.

Regsvr32.exe ctlBlinker.ocx /U

### 18. 3. Майстер ActiveX Control Interface Wizard

Майстер ActiveX Control Interface Wizard спрощує процес написання кода при створенні елемента керування ActiveX. За його допомогою у об’єкта створюються нові властивості, методи, події, а також налагоджуються їх параметри.

Для установки Майстра ActiveX Control Interface Wizard треба вибрати команду **Add-Ins | Add-In Manager**. На екрані з’являється вікно **Add-In Manager**. В списку **Available Add-Ins** виберіть **VB 6 ActiveX Ctrl Interface Wizard**, в групі **Loaded Behavior** установити прапорець **Loaded/Unloaded** і натиснути кнопку

**ОК**. В результаті в меню **Add-Ins** додається команда **ActiveX Control Interface Wizard.**

В майстрі ActiveX Control Interface Wizard є такі діалогові вікна:

□ **Introduction**

□ **Select Interface Members**

□ **Create Custom Interface Members**

□ **Set Mapping**

□ **Set Attributes**

□ **Finished**

**Примітка**

Після роботи майстра можна змінити створюваний ним код, щоб проект повністтю задовольняв ваші потреби.

## 18. 4. Елемент керування ActiveX для біжучого рядка

Текст у біжучому рядку встановлюється значеннями властивості RunningString елемент керування ActiveX, а сам елемент керування буде мати назву **ctlRunningString**.

Щоб створити цей елемент керування, спочатку виконаємо команду **File | New Project**. У вікні **New Project** виберіть позначку **ActiveX Control** і натисніть кнопку **ОК**. Виконайте команду **Project | Properties** і у вікні, що з'явилося, **Project Properties** в полі **Project Name** треба ввести ім'я проекту RunningString. Натиснути кнопку **ОК**. В вікні **Properties** треба встановити об'єкту **UserControl** значення ctlRunningString властивості Name.

Щоб зформувати проект , розташуйте в вікні **UserControl** таймер, графічне поле (**PictureBox**), а в графічному полі, як у контейнері, розташуйте напис (**Label**). В вікні **Properties** встановити значення властивості Caption напису рівним порожньому рядку. Таймер дозволяє переміщувати напис в графічному полі, створюючи ефект біжучого рядка.

Створимо властивість RunningString. Для цього застосуємо майстр ActiveX Control Interface Wizard.

1. Виконайте команду **Add-Ins | Add-Ins Manager**.

З’являється вікно **Add-Ins Manager (Рис.12.8)**, у якому вибираємо **VB 6 ActiveX Ctrl Interface Wizard**. Натискуємо кнопку **ОК**. Отримуємо меню (Рис.12.9)

2. В меню **Add-Ins** з’являється елемент **ActiveX Control Interface Wizard.** Натискуємо на ньому і з’явиться вікно **ActiveX Control Interface Wizard — Introduction** (рис. 12.10.). Це вікно довідкового характеру. Натиснемо кнопку **Next**.

3. З’явиться вікно **ActiveX Control Interface Wizard — Select Interface Members** (рис. 12.11.). В цьому вікні знаходиться список доступних і вибраних властивостей, подій і методів елемента керування. Вони нам не знадобляться і тому треба натиснути на кнопку << для того, щоб очистити список **Select names.** Натисніть кнопку **Next.**

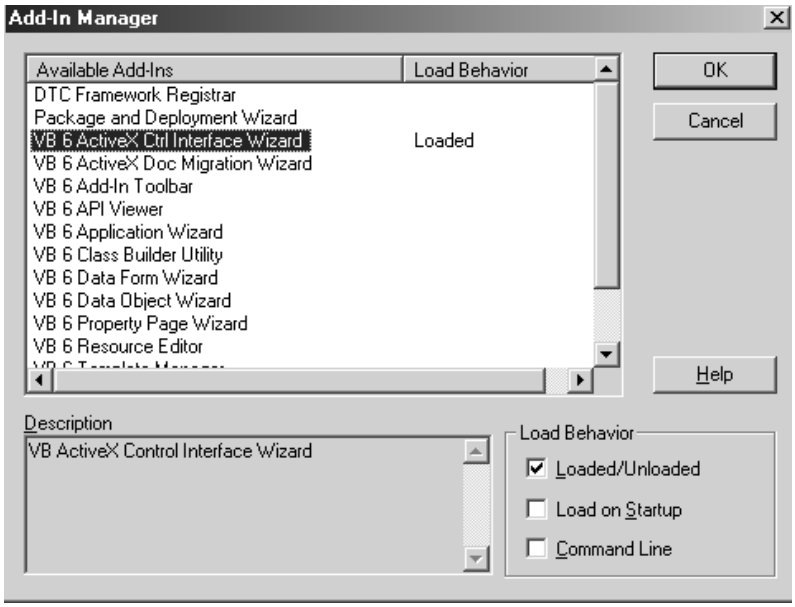



**Рис. 12 .8**. Вікно **Add-Ins Manager**

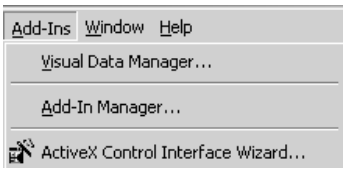



**Рис. 12. 9.** Меню **Add-Ins**

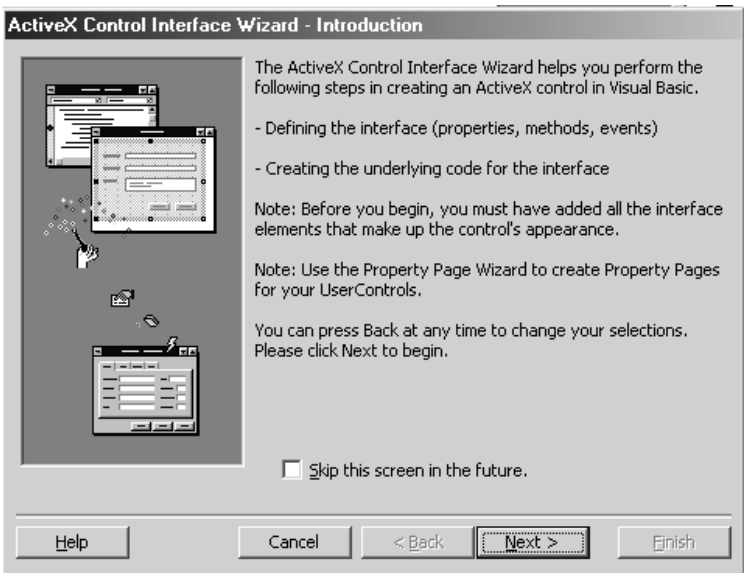



**Рис. 12. 10.** Вікно **ActiveX Control Interface Wizard — Introduction**

З’явиться вікно **ActiveX Control Interface Wizard — Create Custom Interface Members** (рис. 12.12.). Це вікно застосовується для створення нових властивостей, методів і подій елемента управління. Для організації нової властивості треба натиснути кнопку **New**. На екрані з’явиться вікно **Add Custom Member** (рис. 12.13.). В групі **Type** виберіть перемикач **Property**, а в полі **Name** введіть RunningString. Натисніть кнопку **ОК**. Вікно **Add Custom Member** закриється, а в вікні **ActiveX Control Interface Wizard — Create Custom Interface Members** у списку **My Custom Members** з’явиться елемент RunningString. Натисніть кнопку**Next.**

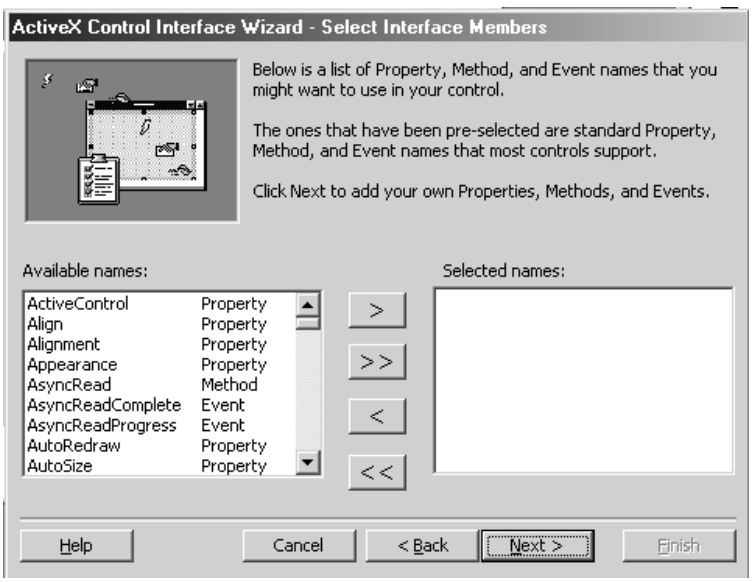



**Рис. 12. 11.** Вікно **ActiveX Control Interface Wizard — Select Interface Members**

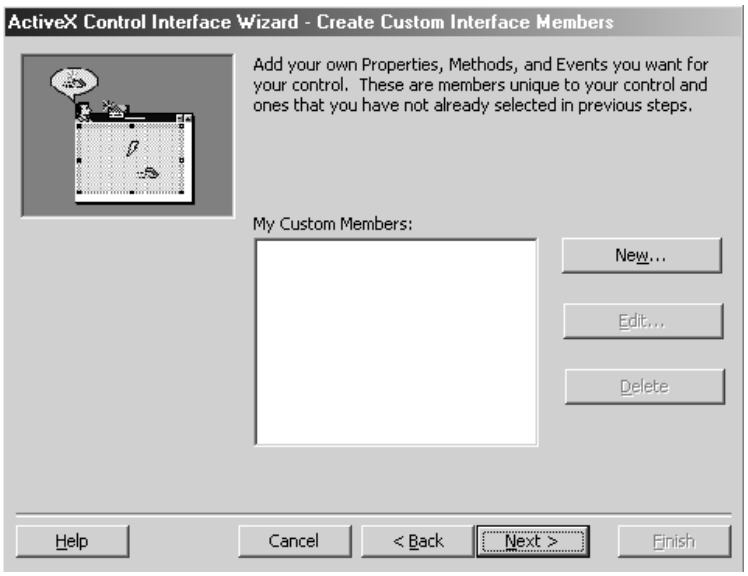



**Рис. 2. 13.** Вікно **ActiveX Control Interface Wizard — Create Custom Interface Members**


**Рис. 2. 14.** Вікно **Add Custom Member**

4. З’явиться вікно **ActiveX Control Interface Wizard — Set Mapping** (рис. 12.14.). Воно призначено для встановлення відповідності між новими

властивостями і властивостями елемента управління, з яких складається елемент, що створюється. В списку **Public Name** виберіть RunningString.

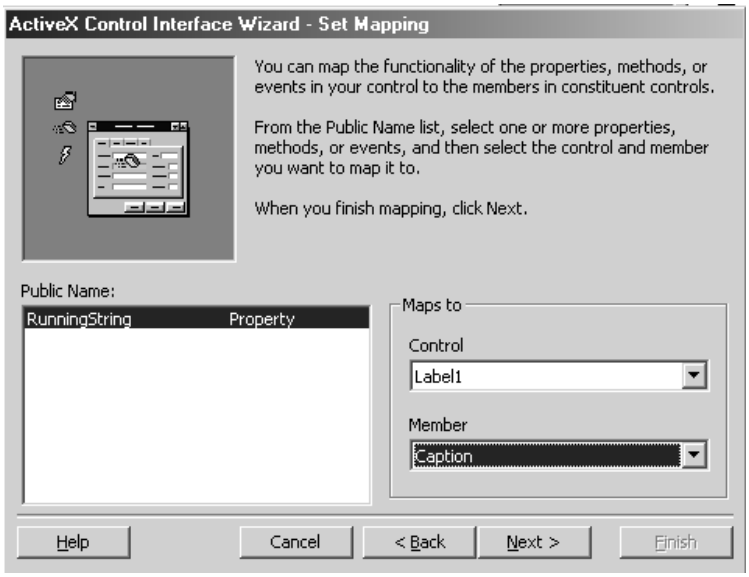



**Рис 12. 14.** Вікно **ActiveX Control Interface Wizard — Set Mapping**

Виберіть у списоку **Control** — Label1, а у списку **Member** — Caption.

Таким чином, встановлення значення властивості RunningString буде еквівалентна встановленню значення властивості Caption надпису Label1, що розташована на об’єкті UserControl. Натисніть кнопку **Next**.

6. З’явиться вікно **ActiveX Control Interface Wizard — Finished!** (рис. 12.15.). Для перегляду звіту про виконану роботу треба встановити прапорець **View Summary Report** і натиснути кнопку **Finish**. З’являється вікно **ActiveX Control Interface Wizard Summary**, у якому дається опис елемента управління, що створений (рис. 12.16.). Майстер автоматично створює в модулі об’єкта UserControl код для елемента управління ctlRunningString.

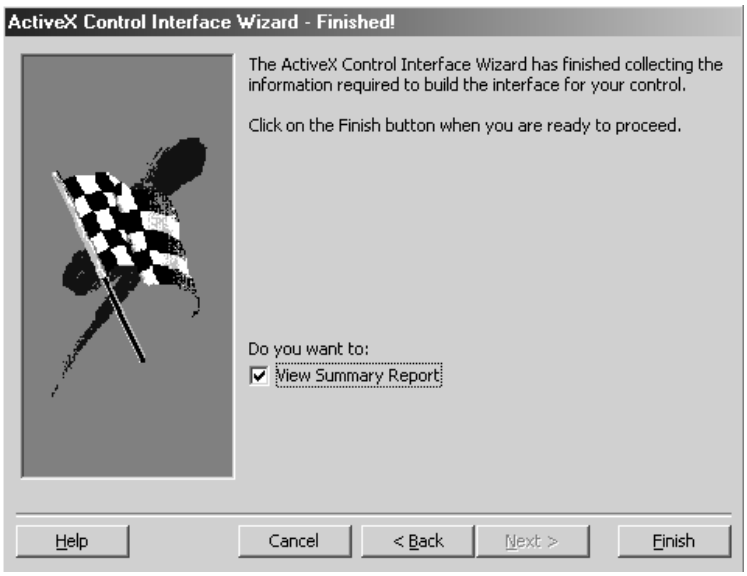



**Рис. 12. 15.** Вікно **ActiveX Control Interface Wizard — Finished!**

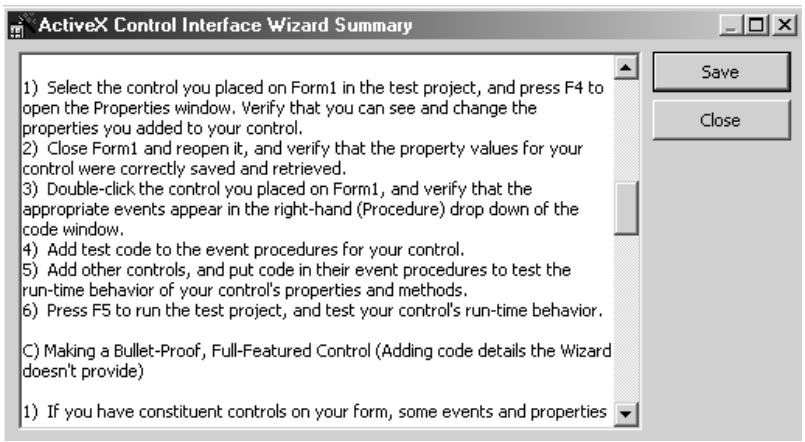



**Рис. 12. 16.** Вікно **ActiveX Control Interface Wizard Summary**

**Код створений майстром для елемента управління *ctlRunningString***

```
'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Label1,Label1,-1,Caption
Public Property Get RunningString() As String
    RunningString = Label1.Caption
End Property

Public Property Let RunningString(ByVal New_RunningString As String)
    Label1.Caption() = New_RunningString
    PropertyChanged "RunningString"
End Property

'Load property values from storage
Private Sub UserControl_ReadProperties(PropBag As PropertyBag)
    Label1.Caption = PropBag.ReadProperty("RunningString", "")
End Sub

'Write property values to storage
Private Sub UserControl_WriteProperties(PropBag As PropertyBag)
    Call PropBag.WriteProperty("RunningString", Label1.Caption, "")
End Sub
```

Для завершення проекту в цей код треба:

- додати процедури, які задають значення властивостей при ініціалізації елемента управління;
- описати, як змінюються розміри і місцезнаходження елементів управління, що містяться в ньому (в даному випадку, написи, таймери і графічні поля) при зміні користувачем розмірів об’єкта-контейнера;
- написати код, який і реалізує потрібний ефект.

В результаті отримуємо код, в якому виділимо код, що створений користувачем.

Повний код елемента управління *ctlRunningString*

```
Private Sub UserControl_Initialize ()
        RunningString = “Andrey Micho”
        Timer1.Interval = 50
End Sub
Private Sub UserControl_Terminate ()
        Timer1.Enabled = False
End Sub
Private Sub UserControl_Resize ()
‘ Графічне поле за своїм розміром співпадає з контейнером
        With Picture1
                .Top = 0
                .Left = 0
                .Width = UserControl.Width
                .Height =UserControl.Height
                .BackColor =vbBlack
        End With
        With Label1
                .Height =UserControl.Height
                .Top = 0
                .Left = 0
                .FontSize 00.8 * Picture1.ScaleY(.Height, vbTwips, vbPoints)
                .AutoSize = True
                .BackColor =vbBlack
                .ForeColor = vbYellow
        End With
If UserControl.Ambient.UserMode Then
        Timer1.Enabled = True
Else
        Timer1.Enabled = False
```

```
End If
End Sub
Private Sub Timer1_Timer ()
        If Ambient.UserMode Then
‘ Напис переміщується, поки не заховається
‘ за лівим краем графічного полю.
‘ Після цього він починає своє переміщення
‘ з правого боку графічного поля і т. д.
        Label1.Move Label1.Left – 30
        If Label1.Left  < -Label1.Width Then
                Label1.Left  = Picture1.Width
        End If
End If
End Sub
‘ Код, що створений майстром
‘ ************************************************************
‘ Попередження! Не вилучайте і не змінюйте
'MappingInfo=Label1,Label1,-1,Caption
Public Property Get RunningString () As String
   RunningString = Label1.Caption
End Property
‘
Public Property Let RunningString(ByVal New_RunningString As String)
   Label1.Caption() = New_RunningString
   PropertyChanged "RunningString"
End Property
‘
‘ Завантаження значень властивостей
Private Sub UserControl_ReadProperties(PropBag As PropertyBag)
   Label1.Caption = PropBag.ReadProperty("RunningString", "")
```

```
End Sub
‘
‘ Записування значень властивостей
Private Sub UserControl_WriteProperties(PropBag As PropertyBag)
   Call PropBag.WriteProperty("RunningString", Label1.Caption, "")
End Sub
```

Збережемо vbp- і ctl-файли, а також відкомпілюємо проект, щоб створити осх-файл після виконання команди *File | Make ctlRunningStr.ocx.

## 18. 5. Процедури обробки подій

Процедури, що обробляють код елемента управління ActiveX, можна поділити на два типи: процедури підтримки і процедури, що задають властивості, методи і події.

*Процедури підтримки*

- Процедура обробки події **InitProperties**, яка задає початкові розміри елемента управління в момент розміщення його на формі при подвійному клацанні на позначці елемента управління із панелі інструментів **ToolBox**, а також початкові значення властивостей.
- Процедура обробки події **Resize** об’єкта **UserControl** задає розміри об’єкта на етапі проектування або виконання. Вона забезпечує зміни розмірів поля уведення і його місцерозташування при зміні розміру елемента управління, що конструюється. З процедурою **Resize** зв’язані властивості **Height, Width** об’єкта **UserControl,** які встановлюють його розміри.
- Процедура обробки події **Terminate** об’єкта **UserControl** описує, що здійснюється при вилученні об’єкта з пам’яті.
- Події об’єкта **UserControl**, які повинні бути запрограмовані, є ReadProperties і WriteProperties. Вони застосовуються для відстежування значень властивостей (в нашому випадку RunningString), які встановлюються на етапі проектування. Це здійснюється властивостями ReadProperties і

WriteProperties об’єкта PropertyBag. Об’єкт PropertyBag використовується для збереження значень властивостей. Впрограмному коді повинні бути дві процедури, в яких для кожної властивості будуть інструкції PropBag.ReadProperty і PropBag.WriteProperty.

*Опис процедур, які задають властивості*

- Процедура Property Let встановлює можливість запису значення властивості. Ця процедура викликається кожний раз при зміні значення властивості або в коді в вікні **Properties**. Коли властивість доступна тільки для читанні, то для нього не треба складати процедуру Property Let. В випадку, коли значення властивості є об’єкт, то використовується процедура Property Let. При встановленні властивостей елемента управління, що входять в об’єкт-контейнер, використовується метод PropertyChanged об’єкта UserControl.
- Процедура Property Get встановлює можливість зчитування значення властивостей. Ця процедура викликається кожний раз, коли код зчитує значення властивості.

Наприклад в коді властивість RunningString реалізується такими двома процедурами.

```
Public Property Get RunningString () As String
   RunningString = Label1.Caption
End Property
‘
Public Property Let RunningString(ByVal New_RunningString As String)
   Label1.Caption() = New_RunningString
   PropertyChanged "RunningString"
End Property
```

## 18. 6. Об’єкт AmbientProperties

Об’єкт AmbientProperties видає інформацію про об’єкт-контейнер UserControl. Переахуємо його найбільш важливі властивості.

| Властивість | Опис |
|---|---|
| UserMode | Логічна властивість, яка повертає False, коли елемент управління знаходиться в режимі конструювання, і True, коли в режимі виконується програма |
| LocaleID | Повертає число, яке відображає використовувану локальну версію Visual Basic |
| DisplayName | Ім'я об'єкта, яке відображається у вікні генерації помилок в час виконання програми |

## 18. 7. Елемент управління ActiveX ctlSpinner

Елемент управління ActiveX ctlSpinner дозволяє за допомогою смуги прокрутки вводити в поле уведення значення. У цього елемента визначені:

- три властивості: Min (мінімальне значення), Max (максимальне значення), Value (поточне значення);
- метод Init, який встановлює поточне значення елемента управління рівним 1;
- подія Change, яка генерує зміну поточного значення елемента управління.

Створимо новий **UserControl**-проект. Виконаємо команду **Project | Properties** і у вікні **Project Properties** в полі **Project Name** введемо ім'я проекту — Spinner. Натиснемо кнопку **ОК**. В вікні **Properties** встановимо для об'єкта **UserControl** значення властивості **Name** рівним **ctlSpinner**.

Розташуємо у вікні **UserControl** поле вводу (Text) і вертикальну смугу прокрутки (VScrollBar). Встановимо значення властивості Text порожнім. За допомогою майстра ActiveX Control Interface Wizard створимо властивості Min, Max, Value і подію Change, які зв'яжемо з властивостями Min, Max, Value вертикальною смугою прокрутки. До згенерованого майстром коду треба додати код, що задає метод Init. В результаті отримуємо код, який створений розробником програми..

**Програмний код:**

```
'Event Declarations:
Event Change() 'MappingInfo=VSCroll,VScroll1, -1,Change
Private Sub UserControl_Initialize()
    Text1.Enabled = False
End Sub
Private Sub UserControl_Resize()
    With Text1
        .Top = 0
        .Left = 0
        .Height = Height
        .Width = 9 * Width / 10
    End With
    With VScroll1
        .Top = 0
        .Left = 9 * Width / 10
        .Height = Height
        .Width = 9 * Width / 10
    End With
End Sub
'
Private Sub VScroll1_Change()
    RasizeEvent Change
    Text1.Text = VScroll1.Value
End Sub
'
Public Sub Init()
Text1.Text = 1
End Sub
' Код, що створений майстром
```

```
'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=VScroll1,VScroll1,-1,Value
Public Property Get Value() As String
    Value = VScroll1.Value
End Property

Public Property Let Value(ByVal New_Value As String)
    VScroll1.Value() = New_Value
    PropertyChanged "Value"
End Property

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=VScroll1,VScroll1,-1,Max
Public Property Get Max() As Integer
    Max = VScroll1.Max
End Property
Public Property Let Max(ByVal New_Max As Integer)
    VScroll1.Max() = New_Max
    PropertyChanged "Max"
End Property
'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=VScroll1,VScroll1,-1,Min
Public Property Get Min() As Integer
    Min = VScroll1.Min
End Property
Public Property Let Min(ByVal New_Min As Integer)
    VScroll1.Min() = New_Min
```

```
    PropertyChanged "Min"
End Property
Private Sub UserControl_ReadProperties(PropBag As PropertyBag)
    VScroll1.Value = PropBag.ReadProperty("Value", "0")
    VScroll1.Max = PropBag.ReadProperty("Max", 32767)
    VScroll1.Min = PropBag.ReadProperty("Min", 0)
End Sub
Private Sub UserControl_WriteProperties(PropBag As PropertyBag)
    Call PropBag.WriteProperty("Value", VScroll1.Value, "0")
    Call PropBag.WriteProperty("Max", VScroll1.Max, 32767)
    Call PropBag.WriteProperty("Min", VScroll1.Min, 0)
End Sub
```

Треба зберегти цей код у vbp- і ctl-файли, а також відкомпілювати проект, створюючи ocx-файл після вибору команди **File | Make ctlSpinner.ocx**.

Перевіримо дію елемента управління **ctlSpinner.** В MS Excel створимо форму. Розташуємо у ній елемент управління **ctlSpinner** і поле уведення. В модулі форми нижче наберемо код тестування. Він забезпечує установку мінімального і максимального значення елемента управління **ctlSpinner** підчас ініціалізації вікна. При прокрутці елемента управління **ctlSpinner** його поточне значення відображається в полі уведення, а при натисненні на формі елемент управління **ctlSpinner** встановлюється в початковий стан.

## 18. 8. Процедури обробки подій і методів

Події елемента управління ActiveX ініціюються із еквівалентних подій або об’єкта UserControl, або із елементів управління, що в ньому містяться. При цьому код складається із оператора об’яви події Event і процедури, в якій повинна знаходитися команда RaiseEvent, яка ініціює подію. Синтаксис:

RaiseEvent eventname [(argumentlist)]

- eventname — ім’я події;

- argumentlist — список параметрів.

Наприклад, ініціювання події Change, що зв'язана з вертикальною смугою прокрутки, в коді забезпечується така інструкція об'яви події і процедури:

```
Event Change () 'Об'ява події
Private Sub Vscroll1_Change ()
    RaiseEvent Change
    Text1.Text = Vscroll1.Value
End Sub
```

Подію Click елемента управління ActiveX можна ініціювати таким чином в коді:

```
Event Click () 'Об'ява події
Private Sub UserControl_Click ()
    RaiseEvent Click
End Sub
```

Методи елемента управління ActiveX в коді створюються звичайними процедурами, які об'являються за допомогою ключового слова Public. Наприклад, в нашому випадку метод Init ініціюється процедурою:

```
Public Sub Init ()
    Text1.Text = 1
End Sub
```

## 18. 9. Модернізація елемента управління

Розглянемо кілька напрямків поліпшення комбінованого списку.

- При натискані клавіші [Enter] значення із поля уведення заноситься у список.
- Колір фона комбінованого списку при отриманні фокусу змінюється, а при втраті — знову стає білим, як було прийнято за замовчуванням.
- При сортуванні елемента списку: по зростанню і по зменшенню, з урахуванням і без урахування регістра.
- При уведенні позачергового значення в полі уведення в списку вибирається найбільш відповідний елемент.

Розглянемо модернізацію елемента управління, додаючи комбінованому списку перші два поліпшення із перерахованих.

Створимо новий проект **ActiveX Control**. Виконаємо команду **Project | Properties** і в вікні **Project Properties** в полі **Project Name** ввести ім'я проекту — Tcombo, а в вікні **Properties** встановити об'єкту UserControl значення ctlTCombo властивості Name.

Зараз можна перейти до створення проекту. Розташуйте у вікні UserControl комбінований список. В вікні **ActiveX Control**. **Interface Wizard — Select Interface Members** знаходиться список доступних і вибраних властивостей, подій і методів елемента управління. Деякі із них нам знадобляться для створення елемента управління, тому за допомогою кнопок > і < залиште в списку **Selected name** такі елементи:

| | | | |
|---|---|---|---|
| AddItem | Enabled | KeyUp | RemoveItem |
| Click | KeyDown | ListCount | Sorted |
| DblClick | KeyPress | LictIndex | Text |

За допомогою вікна **ActiveX Control**. **Interface Wizard — Select Interface Members** майстра створіть:

- властивість EnterColor, який встановлює колір елемента управління при отриманні ним фокусу;
- подія Enter, яка генерується при натисненні клавіші [Enter], коли в полі уведення елемента управління містяться дані (рядок не порожній).

В вікні **ActiveX Control**. **Interface Wizard — Set Mapping** зв'язати властивості, події і методи, які вибрані в вікні **ActiveX Control**. **Interface Wizard — Select Interface Members** з відповідними властивостями, подіями і методами комбінованого поля. Натисніть кнопку **Next**. З'явиться вікно **ActiveX Control**. **Interface Wizard — Set Attributes**. В вікні**ActiveX Control**. **Interface Wizard — Set Attributes** встановлюються:

- тип отримуваного або встановлюваного значення користувацьких властивостей;

- значення властивості, що встановлено за замовчуванням;
- параметри властивості;
- доступність властивості на етапі створення і роботи програми.

Таким чином:

1. В списку **Public Name** виберіть властивість EnterColor.
2. В списку **Data Type** виберіть Long.
3. В поле **Default Name** встановити значення 0.
4. Останні установки вікна **ActiveX Control**. **Interface Wizard — Set Attributes** залиште такими. як вони є. Натисніть кнопку **Next**.

Розглянемо повний код елемента управління **ActiveX ctlCombo**.

Повний код елемента управління **ActiveX ctlCombo**:

```
Enum Spector       'Добавлено
   [Cyan color] = vbCyan
   [Yellow color] = vbYellow
   [White color] = vbWhite
End Enum
'
'Default Property Values:
Const m_def_ListCount = 0
Const m_def_EnterColor = [Yellow color]  'Изменено
'Property Variables:
Dim m_ListCount As Integer
Dim m_EnterColor As Spector  'Изменено
'Event Declarations:
Event Click()
Event DblClick()
Event KeyDown(KeyCode As Integer, Shift As Integer)
Event KeyPress(KeyAscii As Integer)
Event KeyUp(KeyCode As Integer, Shift As Integer)
Event MouseDown(Button As Integer, Shift As Integer, X As Single, Y As Single)
```

```
Event MouseMove(Button As Integer, Shift As Integer, X As Single, Y As Single)
Event MouseUp(Button As Integer, Shift As Integer, X As Single, Y As Single)
Event Enter()
Private Sub UserControl_Initialize()
   With Combo1
      .Clear
      .Top = 0
      .Left = 0
      Width = .Width
   End With
End Sub
Private Sub UserControl_Resize()
   With Combo1
'Графическое поле по своим размерам совпадает с контейнером
      .Width = Width
       Height = .Height
   End With
End Sub
Private Sub Combo1_GotFocus()
   Combo1.BackColor = m_EnterColor
End Sub
Private Sub Combo1_LostFocus()
   Combo1.BackColor = [White color]
End Sub
Private Sub Combo1_KeyDown(KeyCode As Integer, Shift As Integer) 'Изменено
   If KeyCode = vbKeyReturn And Combo1.Text <> Empty Then
      RaiseEvent Enter
      Combo1.AddItem Combo1.Text
   Else
      RaiseEvent KeyDown(KeyCode, Shift)
```

```
    End If
End Sub

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,Enabled
Public Property Get Enabled() As Boolean
    Enabled = Combo1.Enabled
End Property

Public Property Let Enabled(ByVal New_Enabled As Boolean)
    Combo1.Enabled = New_Enabled
    PropertyChanged "Enabled"
End Property
Private Sub Combo1_Clik()
    RaiseEvent Click
End Sub
Private Sub Combo1_DblClik()
    RaiseEvent DblClick
End Sub
Private Sub Combo1_KeyPress(KeyAscii As Integer)
    RaiseEvent KeyPress(KeyAscii)
End Sub
Private Sub Combo1_KeyUp(KeyCode As Integer, Shift As Integer)
    RaiseEvent KeyUp(KeyCode, Shift)
End Sub

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,Sorted
```

```
Public Property Get Sorted() As Boolean
    Sorted = Combo1.Sorted
End Property

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,Text
Public Property Get Text() As String
    Text = Combo1.Text
End Property

Public Property Let Text(ByVal New_Text As String)
    Combo1.Text = New_Text
    PropertyChanged "Text"
End Property

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,ListIndex
Public Property Get ListIndex() As Integer
    ListIndex = Combo1.ListIndex
End Property

Public Property Let ListIndex(ByVal New_ListIndex As Integer)
    Combo1.ListIndex() = New_ListIndex
    PropertyChanged "ListIndex"
End Property

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
```

```
'MemberInfo=7,0,0,0
Public Property Get ListCount() As Integer
    ListCount = m_ListCount
End Property
Public Property Let ListCount(ByVal New_ListCount As Integer)
    m_ListCount = New_ListCount
    PropertyChanged "ListCount"
End Property

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,AddItem
Public Sub AddItem(ByVal Item As String, Optional ByVal Index As Variant)
    Combo1.AddItem Item, Index
End Sub

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MemberInfo=7,0,0,0
Public Property Get ListCount() As Integer
    ListCount = m_ListCount
End Property

Public Property Let ListCount(ByVal New_ListCount As Integer)
    m_ListCount = New_ListCount
    PropertyChanged "ListCount"
End Property
'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,AddItem
```

```
Public Sub AddItem(ByVal Item As String, Optional ByVal Index As Variant)
    Combo1.AddItem Item, Index
End Sub

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MappingInfo=Combo1,Combo1,-1,RemoveItem
Public Sub RemoveItem(ByVal Index As Integer)
    Combo1.RemoveItem Index
End Sub

'WARNING! DO NOT REMOVE OR MODIFY THE FOLLOWING COMMENTED LINES!
'MemberInfo=8,0,0,0
Public Property Get EnterColor() As Spector 'Изменено
     EnterColor = m_EnterColor
End Property

Public Property Let EnterColor(ByVal New_EnterColor As Spector)  'Изменено
    m_EnterColor = New_EnterColor 'Изменено
    PropertyChanged "EnterColor"
End Property

'Initialize Properties for User Control
Private Sub UserControl_InitProperties()
    m_ListCount = m_def_ListCount
    m_EnterColor = m_def_EnterColor
End Sub

'Load property values from storage
```

```
Private Sub UserControl_ReadProperties(PropBag As PropertyBag)
Dim Index As Integer

   Combo1.Enabled = PropBag.ReadProperty("Enabled", Истина)
   m_ListCount = PropBag.ReadProperty("ListCount", m_def_ListCount)
   Combo1.ListIndex = PropBag.ReadProperty("ListIndex", 0)
   Combo1.Text = PropBag.ReadProperty("Text", "Combo1")
   m_EnterColor = PropBag.ReadProperty("EnterColor", m_def_EnterColor)
End Sub

'Write property values to storage
Private Sub UserControl_WriteProperties(PropBag As PropertyBag)

   Call PropBag.WriteProperty("Enabled", Combo1.Enabled, Истина)
   Call PropBag.WriteProperty("ListCount", m_ListCount, m_def_ListCount)
   Call PropBag.WriteProperty("ListIndex", Combo1.ListIndex, 0)
   Call PropBag.WriteProperty("Text", Combo1.Text, "Combo1")
   Call PropBag.WriteProperty("EnterColor", m_EnterColor, m_def_EnterColor)
End Sub
```

## 18. 10. Питання для самоконтролю

1. Як в Visual Basic 6.0. створити середовище розробки елемента керування ActiveX?
2. За допомогою якої процедури можна виводити текст, змінюючи його колір?
3. Як діє властивістю таймера Enabled?
4. Чи елемент керування Timer відображається в вікні в час виконання програми?
5. Де можна розташувати таймер на формі?
6. Які команди треба виконати для збереження проекту?

7. Як створити осх-файл?

8. Як зробити перевірку роботи?

9. Чи треба заєреструвати елемент керування ActiveX (занести його в реєстр) для використання на ПЕОМ?

10. Як виконується програма VB при компіляції конструйованих елементів керування ActiveX?

11. Чи треба забезпечити реєстрацію елементів керування ActiveX при їх розповсюджені?

12. Яка програма дозволяє створити із побудованого проекту інсталяційний пакет з програмою установки Setup.exe, при запуску якої здійснюється автоматична реєстрація на ПЕОМ користувача?

13. Що здійснює майстер ActiveX Control Interface Wizard?

14. Як встановити майстер ActiveX Control Interface Wizard?

15. Які діалогові вікна є в майстрі ActiveX Control Interface Wizard?

## 19. САМОСТІЙНА РОБОТА

Самостійна робота по вивченню запропонованого матеріалу лекцій полягає в досягненні можливості відповісти на запитання, що поданні у кінці кожної лекції (питання для самоконтролю).

Для контролю знань за визначеним темам для заочної форми навчання подані варіанти виконання контрольних робіт у вигляді таблиць, лабораторних та самостійних робіт.

## 19. 1. Контрольна робота № 5 за темами 12, 13, 14

В усіх варіантах контрольних робіт дати письмові відповіді на питання

| Варіант | Питання параграфа 12.10. | Питання параграфа 13.3. | Питання параграфа 14.9. |
|---|---|---|---|
| 1 | 1, 16, 33 | 1, 16 | 1, 16 |
| 2 | 2, 17, 34 | 2, 17 | 2, 17 |
| 3 | 3, 18, 35 | 3, 18 | 3, 18 |
| 4 | 4, 19, 37 | 4, 19 | 4, 19 |
| 5 | 5, 20, 38 | 5, 9 | 5, 20 |
| 6 | 6, 21, 33 | 6, 10 | 6, 21 |
| 7 | 7, 22, 32 | 7, 11 | 7, 22 |
| 8 | 8, 23, 31 | 8, 12 | 8, 23 |
| 9 | 9, 24, 29 | 9, 13 | 9, 24 |
| 10 | 10, 23, 25 | 10, 14 | 10, 25 |
| 11 | 11, 22, 26 | 11, 15 | 11, 26 |
| 12 | 12, 14, 27, | 12, 16 | 12, 27 |
| 13 | 13, 28, 31 | 13, 17 | 13, 28 |
| 14 | 14, 17, 29 | 14, 18 | 14, 29 |
| 15 | 15, 18, 30, | 15, 19 | 15, 30 |

### Контрольні запитання:

1. Яким значенням властивості елемента управління ActiveX встановлюється текст в біжучому рядку?
2. Яку роль відіграє таймер при створенні ефекту біжучого рядка?
3. Як створити властивість RunningString?
4. Для чого використовується вікно ActiveX Control Interface Wizard — Select Interface Members?
5. Для чого використовується вікно ActiveX Control Interface Wizard — Create Custom Interface Member

## 19. 2. Контрольні роботи за темами 16, 17, 18

| Варіант | Питання параграфа 16.4. | Питання параграфа 17.3. | Питання параграфа 18.10. |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 |
| 3 | 3 | 3 | 3 |
| 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | 5 |
| 6 | 6 | 6 | 6 |
| 7 | 7 | 7 | 7 |
| 8 | 8 | 8 | 8 |
| 9 | 9 | 9 | 9 |
| 10 | 10 | 10 | 10 |
| 11 | 11 | 11 | 11 |
| 12 | 12 | 12 | 12 |
| 13 | 13 | 13 | 13 |
| 14 | 14 | 14 | 14 |
| 15 | 15 | 15 | 15 |

# РОЗДІЛ 3. ОСНОВИ МЕРЕЖНИХ ТЕХНОЛОГІЙ ТЕЛЕКОМУНІКАЦІЇ ТА РОБОТИ З БАЗАМИ ДАНИХ

## 20. Апаратні засоби зв’язку в комп’ютерних мережах

### 20.1. Характеристики передавального середовищв в комп'ютерних мережах

ЛОМ можна створювати з використанням різних типів кабелю. Найдешевшим є кабель с*кручена пара* (пара скручених ізольованих мідних проводів, що використовується в телефонії. Він може бути *екранованим* і *неекранованим*. Екранований більш стійкий до електромагнітних завад. Однак на практиці частіше використовується неекранований кабель, тому що такий тип кабелю використовується для розведення телефонних ліній і він дешевше екранованого. Що найкраще підходить для малих установ. *Недоліками* даного кабелю є високий коефіцієнт згасання сигналу і висока чутливість до електромагнітних завад, тому максимальна відстань між активними пристроями в ЛОМ при використанні скрученої пари - до 100 метрів.

*Коаксіальний кабель*. Цей кабель може використовуватися в двох різних системах передачі даних: без модуляції сигналу і з модуляцією. У першому випадку цифровий сигнал використовується в тому вигляді, у якому він надходить із ПК і відразу ж передається по кабелю на прийомну станцію. Він має один канал передачі зі швидкістю до 10 Мбіт/сек і максимальний радіус дії 4000 м. В другому випадку цифровий сигнал перетворюють в аналоговий і направляють його на прийомну станцію, де він знову перетворюється в цифровий. Операція перетворення сигналу виконується модемом (модулятор/демодулятор); кожна станція повинна мати свій модем. Цей спосіб передачі є багатоканальним (забезпечує передачу по десятках каналів,

використовуючи для цього всього лише один кабель). Таким способом можна передавати звуки, відеосигнали, дані. Довжина кабелю може досягати 50 км.

Передача сигналу з модуляцією більш дорога, ніж без модуляції. Тому, найбільш ефективне його використання при передачі даних між великими підприємствами.

*Оптоволоконий кабель* є новітньою технологією, використовуваною в ЛОМ. Носієм інформації є світловий промінь, що модулюється мережею і приймає форму сигналу. Така система стійка до зовнішніх електричних перешкод і в такий спосіб можливе дуже швидке і безпомилкове передавання даних (до 2 Гбит/с), і вона забезпечує таємність переданої інформації. Оптоволоконні кабелі виготовляють одномодовими (один промінь у волокні) з товщиною волокна ~10 мкм. та багатомодові (кілька променів у волокні під різними кутами до вісі волокна) з товщинами 50,62.5, 199 та 140 мкм. Джерелом світла найчастіше буває напівпровідниковий лазер на хвилях 1.3 та 1.55 мкм. (смуга перепускання досягає 2 ГГц), або суперлюмінесцентний діод на хвилях 1.3 та 0.85 мкм. (смуга перепускання- до 900 МГц). Кількість каналів у таких кабелях величезна. Передача даних виконується тільки в симплексному режимі, тому для організації обміну даними пристрою необхідно з'єднуватися двома оптичними волокнами (на практиці оптоволоконий кабель завжди має парне число волокон). *До недоліків* можна віднести велику вартість, а також складність приєднання, обмежену гнучкість кабеля та інше..

Радіохвилі в мікрохвильовому діапазоні використовуються як передавальне середовище в б*езпровідних локальних мережах*, або між мостами чи шлюзами для зв'язку між ЛОМ. У першому випадку максимальна відстань між станціями складає 200-300 м, у другому — це відстань прямої видимості. Швидкість передачі даних — до 2 Мбіт/с.

Безпровідні локальні мережі (БЛМ) вважаються перспективним напрямком розвитку ЛМ. Їхня *перевага* — простота і мобільність. Зникають проблеми, зв'язані з прокладкою і монтажем кабельних з'єднань. Досить установити інтерфейсні плати на робочі станції і мережа готова до роботи. Стримуючим

фактором широкого розвитку БЛМ є відсутність стандарту для таких мереж. Існуючі БЛМ, виконані різними фірмами, як правило, цілком несумісні між собою.

Тому необхідно дочекатися прийняття й опублікування стандарту IEEE 802.11 (Розробка стандартів в області локальних і регіональних мереж).

Для встановлення зв'язку окремих ПК з ЛМ або зв'язку між окремими ЛМ (регіональні мережі) найчастіше використовують телефонні лінії та супутникові канали зв'язку (глобальні мережі). При передаванні даних на великі відстані враховуються такі особливості передавального середовища як згасання і спотворення сигналу. Для боротьби із згасанням використовуються підсилювачі для аналогового сигналу та повторювачі –для імпульсного (цифрового), розташовані на певних відстанях між ними. Спотворення сигналу, викликане проходженням його через передавальне середовище і необхідні пристрої, різне для аналогового та цифрового сигналів. Цифровий (імпульсний) сигнал спотворюється значно більше ніж аналоговий. Саме тому при передаванні на великі відстані використовують модульований (аналоговий) сигнал, укий отримують з допомогою модема.

## 20.2. Передавання даних з використанням адаптера.

- *Вибір і під'єднання адаптера*

Адаптери станцій локальної мережі безпосередньо приєднують до внутрішньої шини введення-виведення персонального комп'ютера (ПК). Вони дають змогу досягти значно більших швидкостей передавання ніж через послідовний та паралельний порти. В адаптерах апаратно реалізовані, як правило, протоколи фізичного та канального рівнів.

Адаптери виготовляють різні фірми і для різних мережевих технологій. Тому при вибиранні адаптера треба знати такі характеристики:

- До якої мережі він належить (*Ethernet. Token Ring, FDDI,….*);

- Яку має розрядність (8.16,32) та до якої шини приєднується (ISA, EISA, PCI, NCA…);
- Яку має потужність та які алгоритми використовує (для робочої станції чи сервера);
- Які роз'єднувачі (BNC- тонкий коаксіальний кабель,AUI – товстий коаксіальний кабель, R145 – скручена пара, MIC, ST,SC- волоконнооптичний кабель) використовуються.

♦ *Будова та складові частини адаптера.*

Як приклад розглянемо адаптер №15210 для мережі *Ethernet* (виробник-фірма Interlan). До його складу входчять:

- співпроцесор INTEL 82586, який виконує деякі функції опрацювання інформаційних кадрів протоколу канального рівня. Співпроцесор (СП) кодує інформацію перед передаванням в мережу (декодує після приймання), виявляє та виправляє помилки, виконує головні функції з реалізації MAC-підрівня (доступ до мережі);
- оперативна пам'ять (8Кб) У цю пам'ять записують інформацію перед передаванням і після передавання її може читати одночасно і центральний процесор (ЦП), і співпроцесор;
- роз'єднувач розширення- для приєднання додаткової мікросхеми пам'яті;
- вісім регістрів стану та керування- вони дають змогу ЦП та СП обмінюватися командами;
- ПЗП адреси- містять унікальну мережеву адресу комп'ютера, встановлену виробником (48 біт у *Ethernet*);
- перемикачі – для конфігурування адаптера;
- кабельні роз'єднувачі – для приєднання до мережі; трансівер (приймач-передавач) для роботи тонким *Ethernet*;
- роз'єднувач для приєднування до системної шини ПК.

♦ *Робота адаптера.*

При передаванні даних комунікаційне ПЗ будує кадри *Ethernet* та записує їх у пам'ять адаптера. У регістри керування та стану записується команда на

передавання кадру, адреса та кількість інформації для передавання. Мережевий СП аналізує значення регістрів, бере кожен кадр, опрацьовує його згідно з вимогами протоколу і передає у мережу.

При прийманні мережевий СП постійно стежить через трансівер за кадрами в мережі та виділяє ті, які призначені для конкретного адаптера. ПЕревіряє правильність даних, розміщує їх у пам'яті і видає ЦП –у переривання з визначеним номером. ЦП та комунікаційне ПЗ відкидає службові дані, аналізує прийняті дані та переміщує їх у головну пам'ять.

Конфігурування адаптера (визначення: адреси пам'яті, куди відображаються регістри стану та керування; адреси пам'яті, куди відображається внутрішня пам'ять адаптера; номера переривання, за яким ЦП повідомляють про прийняті дані) в розглянутому прикладі здійснюється за допомогою перемикачів. В більш сучасних адаптерах конфігурування здійснюється за допомогою програм, що додаються до адаптера, або засобами операційної системи.

## 20.3. Способи використання модемів

В залежності від типу передавального середовища розрізняють такі модеми:

- ♦ Телефонні модеми. Це найдешевші і найбільш вживані модеми, що використовуються:
  - для під'єднання віддаленого комп'ютера до мережі (тут він відіграє роль повільного адаптера, бо під'єднується через паралельний порт згідно стандарту RS-232C, розробленому в 1969 р.);
  - для дистанційного керування. Тут через модем з віддаленого комп'ютера передається тільки інформація керування, а опрацьовуються дані комп'ютером мережі. Клавіатура та екран

віддаленого комп'ютера працюють паралельно з клавіатурою і екраном комп'ютера мережі;

- для зв'язку між мережами. Тут використовується сервер доступу-спеціалізований пристрій, приєднаний до ЛМ, що має свій власний процесор або кілька процесорів і виконує запити, які надходять з модемів. У випадку багатопроцесорного сервера доступу існує два варіанти: 1) окремий модем під'єднується до окремої плати з процесором, а під'єднування до мережі здійснюється з допомогою одного адаптера; 2) кожна плата з модемом і процесором під'єднується до мережі через окремий адаптер.Однопроцесорним сервером доступу може бути звичайний ПК, що працює в режимі розподілу часу і опрацьовує окремі виклики та приєднаний до мережі через адаптер.

Телефонні модеми бувають двопровідні (симплексний зв'язок) та чотирипровідні (дуплексний зв'язок). Виготовляються телефонні модеми у вигляді внутрішніх пристроїв ПК та у вигляді настільних приладів. Існує великий набір модемів, що різняться швидкістю передавання і, відповідно, цінами.

- ♦ Радіомодеми. Вони мають вбудований радіоприймач/передавач. Використовуються при передаванні даних радіоканалами;
- ♦ Стільникові модеми – для передавання в стільниковій мережі;
- ♦ Оптичні модеми- для модуляції світлового променю.

За функціональним призначенням модеми поділяються на такі групи:

- ♦ Модеми широкого вжитку (для під'єднування телефонними лініями окремих ПК до мережі ( в тому числі і до Інтрернет). Вони відносно дешеві і доступні для широкого кола користувачів;
- ♦ Мережеві модеми. Вони мають вбудований адаптер локальної мережі і працюють як вузол ЛМ. Виготовляються комплектно. Досить дорогі. Вони забезпечують як доступ до абонентів мережі, так і мережеві і міжмережеві зв'язки. Існують так звані *модем-сервери*. Вони обслуговують виклики, що виходять з мережі, ведуть облік часу використання зовнішніх каналів

зв’язку. Поєднання функцій *сервера доступу* і *модем-сервера* реалізується у *сервері асинхронного зв’язку*;

- Модеми для швидкого передавння прямим кабелем (використовується не телефонний канал, а пряме провідникове сполучення) В цьому разі досягається швидкість передавання до 2Мбіт/с на відстань до 15 км.

## 20.4. Інтелектуальні засоби сполучення в мережах

До засобів сполучення в мережах відносяться наступні пристрої, ступінь інтелектуальності яких зростає в порядку їх розміщення в списку, це: повторювачі, концентратори, мости, комутатори, маршрутизатори, шлюзи. Ступінь інтелектуальності визначається внеском програмних засобів у функціонування пристрою. Чим більш автоматизовано функціонування пристрою, тип вищий рівень його функціонування в семирівневій моделі зв’язку.

Усі перелічені пристрої в різних мережевих технологіях дещо відрізняються. Тому розглянемо лише найбільш загальні призначення цих пристроїв, маючи на увазі, що їх функції можуть значно перекриватись між собою навіть в одній і тій хе мерехевій технології. Таким чином, розрізняють:

- Повторювачі (repeaters) – підсилювачі дискретних сигналів. Використовуються для розширення мережі, функціонують на фізичному рівні;
- Концентратори (hubs) – функціонально близькі до повторювачів. Використовуються у локальних мережах з топологією зірка та з деревоподібною топологією. Кожному порту концентратора відповідає окрема робоча станція. Перемикати РС можна вручну і програмно. Концентратор працює на фізичному рівні;
- Мости (bridges) – це апаратно-програмні блоки, які дають змогу сполучати локальні мережі з різним середовищем передавання та з різними мережевими технологіями. Вони працюють на канальному рівні і аналізують адреси в кадрах. Головна функція мосту – фільтрування кадрів між приєднаними сегментами і, як наслідок, зменшення їх навантаження. Міст повинен вирішувати такі проблеми як незбіжність форматів кадрів, різниця в

перепускній здатності та завантаженності, максимальний розмір кадру сполучуваних систем. Якщо раніше міст розглядався як пристрій для з'єднування двох однакови за технологією мереж, то сучасні мости можуть з'єднувати кілька мереж і з різними технологіями;

- Комутатор (switch) – пристрій, функціонально близький до мосту, виконаний у вигляді мережевого концентратора, що працює як швидкий багатопортовий міст. Вбудований механізм комутації дає змогу виконувати сегментування локальної мерехі та виділяти смугу перепускання станціям мережі. Розрізняють два підходи до створення мостів – один -у мережі *Ethernet*, другий – у мережі *Token Ring*. Міст у мережі *Ethernet* називається прозорим. Він об'єднує мережі автоматично при приєднанні кабелів. Прозорий міст приймає всі кадри і спрямовує в кокретний вихідний порт на підставі таблиці адрес, яка ставить у відповідність усі відомі мосту адреси призначеним лініям (портам), куди треба спрямувати кадр.Таблиці створюються автоматично. При зміні топології чи переміщенні комп'ютера відбувається автоматичне налаштування мережі.

При спрямуванні кадру діють такі правила:

якщо відправник і одержувач знаходяться в одній ЛМ, кадр відкидається;

якщо відправник і одержувач знаходяться в різних мережах, кадр спрямовується у вихідний порт згідно адресної таблиці;

якщо визначити вихідний порт не вдається, кадр розсилається на всі порти крім того, на який він надійшов.

Міст *Token Ring* – це міст з визначенням шляху за запитом джерела. Головні функції маршрутизаціївиконують не мости, а робочі станції. Відправник кадру повинен знати, що одержувач знаходиться в іншій мережі. Мережі і мости ідентифікуються унікальними кодами. Функція мосту полягає в опрацюванні послідовності номерів шляху: міст,мережа,міст,мережа….Якщо станція відправник не знає шляху до одержувача, то вона надсилає розшуковий кадр. Цей кадр передають усі станції і в результаті цей кадр потрапляє до всіх

станцій. Відповідь також розсилається циркулярно. Кожний міст на шляху кадру записує в нього інформацію про час руху кадру і про себе.

Недоліком прозорих мостів є циркулярне розсилання кадрів, коли немає зворотніх передавань.

Недоліком мостів *Token Ring* є експоненціальне збільшення розшукових кадрів у складних мережах. Аналізуючи використання мостів як комутаторів можна зробити певні висновки:

міст як комутатор дає змогу реалізувати технологію комутації локальних мереж;

міст надає адміністратору додаткові засоби фільтрування потоку між групами станцій, що підвищує безпеку системи;

за допомогою моту можна сполучати декілька мереж різного типу або з різним середовищем передавання;

міст дає змогу не обмежувати кількість сегментів у ЛМ.

Мости доцільно застосовувати у невеликих і середніх мережах. Зі збільшенням мережі доводиться використовувати маршрутизатори.

- Маршрутизатори (routers)-це апаратно-програмні пристрої, які дають змогу сполучати різні ЛМ та виконують як і мости, фунуцію маршрутизації. Однак, на відміну від мостів, маршрутизація в них виконується на мережевому рівні. Кожен порт маршрутизатора має свою канальну та мережеву адреси, як і робоча станція. Станція, яка хоче передати пакет у зовнішню мережу, формує кадр з адресою порту марщрутизатора і надсилає його. Тому маршрутизатор опрацьовує не весь потік інформації, а тільки кадри, що адресовані безпосередньо йому. Далі маршрутизатор працює з пакетом мережевого рівня, аналізуючи мережеву адресу. На підставі мережевої адреси і внутрішніх таблиць маршрутизатора пакет буде спрямовано через інший порт до наступного вузла мережі за оптимальним маршрутом.

У статичних маршрутизаторах використовуються постійні таблиці маршрутизації, які можна змінювати вручну (адміністратором). У динамічних – таблиці маршрутизації створюються автоматично на основі аналізу ефективності і надійності маршруту.

Маршрутизатори ефективніше використовують канали зв’язку ніж мости, але набагато дорожчі (вони й більш інтелектуальні).

♦ Шлюз (gateway) –це машина, або порт колективного доступу, який об’єднує кілька мереж, або використовується для приєднання мережі до головної ЕОМ (main frame)/У цьому випадку, якзвичайно, перетворюються формати даних. Тому шлюз прац.є з протоколами верхніх рівнів (сеансовий, відображення, прикладний). Реалізація шлюзів значно дорожча, проте і інтелектуальні можливості їх більші ніж у мостів та маршрутизаторів.

Серед інтелектуальних засобів зв’язку особливе місце займають модеми та сервери. Якщо розглянуті тут пристрої в певній мірі являються допоміжними засобами, то такі пристрої як адаптери являються невід’ємними елементами системи зв’язку в мережах. Їх доцільно розглядати окремо і більш детально.

## 20.5. Способи комутації в комп’ютерних мережах.

*Комутаційне середовище* містить у собі безліч серверів і ЕОМ, з'єднаних фізичними (магістральними) каналами зв'язку, що використовують телефонні, коаксіальні кабелі, супутникові канали зв'язку. Обчислювальні мережі за способом передавання інформації поділяються на мережі з комутацією каналів, мережі з комутацією повідомлень, мережі з комутацією пакетів і інтегральні мережі. Кожний з цих методів має свої переваги і недоліки. *Перевагою* мереж з комутацією каналів є простота реалізації (пряме з'єднання), а *недоліком* — низький коефіцієнт використання каналів, висока вартість передачі даних, підвищення часу чекання інших користувачів. При комутації повідомлень передача даних (повідомлення) здійснюється після звільнення каналу, поки воно не дійде до адресата. Кожен сервер робить прийом, перевірку, зборку, маршрутизацію і передачу повідомлення. *Недоліком* даного способу є низька швидкість передачі інформації, неможливість ведення діалогу між

користувачами. До переваг можна віднести — зменшення вартості передавання, прискорення передавання. Пакетна комутація має на увазі обмін невеликими пакетами (частина повідомлення) фіксованої структури, що не створює черг у вузлах комутації. *Перевага*: швидке з'єднання, надійність, ефективність використання мережі. При даному методі проблема передавання пакета зважується способом фіксованої маршрутизації. Вона припускає наявність таблиці маршрутів, де закріплено маршрут від одного користувача до іншого. Мережі, що здійснюють комутацію каналів, повідомлень і пакетів, називаються інтегрованими. До таких мереж відноситься розроблена останнім часом нова мережна технологія АТМ.

*АТМ* — це комунікаційна технологія, що поєднує принципи комутації пакетів і каналів для передачі інформації різного типу. АТМ — (асинхроний режим передачі), дана технологія передбачає інтегровану передачу мови, даних і відеоінформації в єдиному цифровому вигляді по одному й тому ж каналу зв'язку. Це дозволяє відмовитися від твердих обмежень на надану користувачу смугу перепускання каналу зв'язку, відмовитися від поділу каналів по типах інформації для передавання і значно розширити коло наданих послуг. Основною *перевагою* нової технології є відсутність орієнтації на який-небудь тип інформації. Поєднувані в рамках АТМ інформаційні потоки від джерел інформації різної природи різко відрізняються один від одного вимогами до смуги перепускання. Якщо дані у ЛОМ у більшості випадків не вимагають гарантованого часу доставки пакетів і, відповідно, сталості смуги перепускання каналу зв'язку, то системи кабельного телебачення і передачі мови в інтерактивному режимі без виконання цієї умови немислимі. Тому процедура встановлення з'єднання в АТМ-мережі передбачає попереднє визначення типу інформації для передавання, необхідної смуги перепускання і пріорітет на заняття каналу зв'язку, що мінімізує завантаження міжвузловых каналів зв'язку і забезпечує надання послуг із заданою якістю.

Головною відмінністю АТМ від існуючих технологій передавання інформації є її висока швидкість — до 10 Гбіт/с на канал зв'язку. (На

сьогоднішній день — 2,5 Гбіт/с). АТМ об'єктивно сполучає функції, виконувані локальними і глобальними мережами. Віддаленим користувачам надається «прозорий» доступ до будь-яких загальних інформаційних ресурсів, а також забезпечується все різноманіття послуг глобальних телекомунікацій. Дана особливість технології АТМ робить її незамінною при створенні інтегрованих розподілених корпоративних інформаційних мереж на базі волоконо-оптичних каналів зв'язку. Крім того, ефективними рівнями застосування АТМ є високошвидкісні ЛОМ зі специфічними вимогами до трафіку (підтримуючого відео- і CAD/CAM-файли), а також магістральні й абонентські канали передавання в регіональних і внутріміських широкосмугових мережах з інтеграцією обслуговування.

Основною відмінністю АТМ від традиційних ЛОМ-технологій є те, що АТМ за своєю природою орієнтована на установлення віртуальних з'єднань. *Віртуальне з'єднання* — це сконфігуроване певним чином середовище між двома чи більше кінцевими пристроями для передавання інформації. *Віртуальний канал* — фіксований маршрут, що складається з послідовності номерів портів комутаторів, через які проходять усі пакети (комірки) при даному сеансі зв'язку від одного користувача до іншого. Віртуальні канали завжди однонапрямлені, тобто для передачі в зворотному напрямку між тими ж користувачами використовуються вже інші номери ідентифікаторів. Поняття віртуального шляху використовується на якій-небудь ділянці мережі: кілька віртуальних каналів можуть проходити по одній і тій же фізичній ланці, що дає можливість комутатору переключати цілі групи віртуальних каналів. Кожен фізичний канал може містити кілька віртуальних шляхів і каналів. Поза як конфігурація віртуальних з'єднань не зв'язана з фізичними каналами, то топологія АТМ мережі може бути будь-якою. Комутатори при цьому можуть бути з'єднані в шину, чи кільце, чи зірку, але частіше — це суміш усіх можливих з'єднань. Це дає можливість реалізовувати резервування зв'язків, що підвищує надійність мережі.

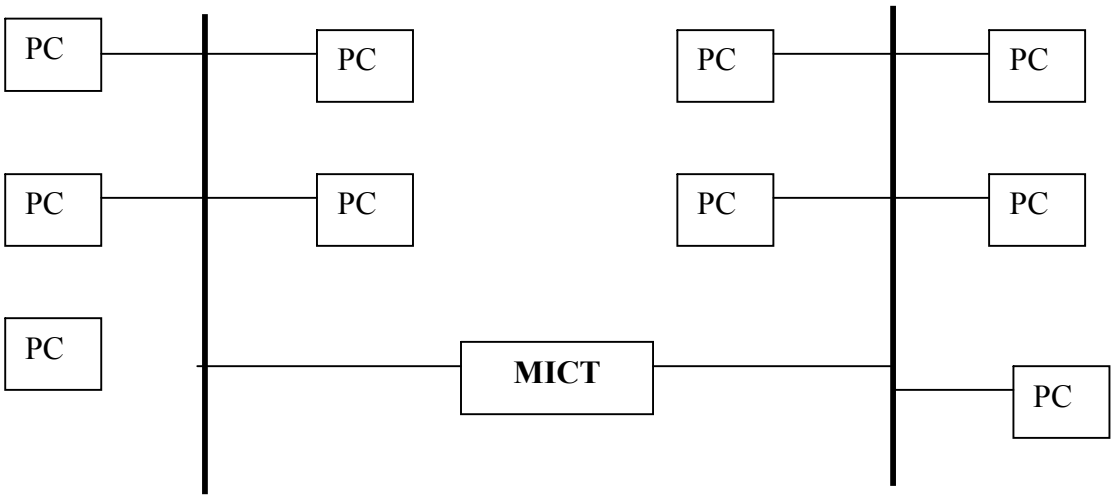



**Рис. 20. 1.** Приклад розділення великої мережі

Звичайні локальні мережі (Ethernet, Token Ring) не перевіряють доступність пристрою призначення, а просто посилають туди пакет з інформацією. Пакет повинен мати адресу призначення, що перевіряється мережними пристроями на відповідність зі своєю власною адресою. Перед передачею яких-небудь повідомлень в АТМ станцію — джерело перевіряє доступність станції призначення і, тільки після цього, встановлюється з'єднання. Тільки цим двом станціям видний потік інформації.

АТМ реалізує комутацію коротких пакетів (комірок), накладену на комутацію віртуальних каналів. На відміну від звичайних інформаційних пакетів комірки не містять адресної інформації і контрольної суми. Комутація відбувається на основі ідентифікатора віртуального каналу, що визначає одне з організованих з'єднань. Контрольна сума вважається непотрібною через використання високоякісної кабельної системи з малою імовірністю помилки. АТМ орієнтовано на з'єднання протоколом. Перед передачею інформації між користувачами організується віртуальний чи логічний канал зв'язку, що залишається в їхньому розпорядженні до закінчення взаємодії. Параметри цього каналу можуть бути різними, у залежності від виду трафіку і його інтенсивності.

Для передачі звуку визначається тільки потрібна фіксована смуга перепускання, а для файлового обміну між комп'ютерами даються параметри середньої і максимальної інтенсивності трафіку. Тому що комірки мають постійну довжину (53 байта), затримки в надходженні нової інформації до споживача завжди однакові. АТМ-комірки легко обробляються при проходженні через комутатор. При обробці пакета маршрутизатор спочатку цілком його приймає в буфер, перевіряє контрольну суму, аналізує адресну інформацію, зміст поданих даних і, тільки після цього, відправляє даний пакет. Програми сучасних маршрутизаторів містять до декількох мільйонів рядків коду, звідси дорожнеча таких пристроїв. На відміну від них комутатор АТМ вирішує свої задачі апаратним шляхом. Комутатор, прочитавши ідентифікатор у заголовку комірки, переправляє її з одного порту в інший, не задумуючись про її зміст.

Виходячи з вищесказаного можна зробити наступні висновки:

- мережа АТМ має завжди більшу перепускну здатність, чим сума всіх реалізованих віртуальних каналів. При цьому контроль здійснюється за рахунок обмеження підключення до мережі нових користувачів логічними засобами самої мережі;
- керування потоком даних здійснюється кінцевим устаткуванням; сама АТМ мережа не має власних засобів для цього;
- на фізичному рівні помилки практично відсутні. АТМ мережа не має механізму перевірки помилок і їхніх виправлень;
- відсоток загублених комірок дуже невеликий і передбачуваний. АТМ не може функціонувати на ненадійних каналах.

Існуючі в наш час телекомунікаційні системи страждають такими недоліками:

- залежність від виду інформації, що вони транспортують;
- відсутність гнучкості, тому що сучасні телекомунікаційні системи практично не забезпечують адаптацію до змін вимог з боку систем керування, до обсягів переданої інформації, до швидкості передавання, часу доставляння і вірогідності;

- низька ефективність використання ресурсів.

Останнім часом з'явилася можливість створення на базі технології АТМ єдиної телекомунікаційної системи — широкосмугової цифрової мережі інтегрального обслуговування (ШЦМІО), що забезпечить виконання наступних функцій:

- транспортування усіх видів інформації за допомогою єдиного асинхроного методу переносу (АТМ), при якому кожен користувач одержує від мережі тільки той ресурс, що йому необхідний;
- підтримку інтерактивних служб і служб розподілу інформації з виконанням вимог як до імовірності блокування, так і вчасного доставлення інформації;
- підтримку режимів із встановленням і без установлення з'єднання між абонентами;
- передавання як безперервного, так і поблокового трафіку, що за рахунок мультиплексування дозволяє більш ефективно використовувати єдині мережні ресурси;
- перетворення сигналів і повідомлень усередині мережі на базі цифрової обробки сигналів;
- забезпечення користувачів такими послугами, як телекерування і телеконтроль, відеотелефон, високошвидкісне передавання даних, надання даних і відеоінформації за вимогою.

З кожним днем росте інтерес до впровадження в телекомунікаційні мережі технології АТМ, особливо маючи на увазі такі фактори як:

- розвиток систем віддаленої обробки даних, що вимагають передавання досить великих обсягів інформації практично в реальному масштабі часу;
- безупинний ріст вимог до високошвидкісних трактів, що поєднують ЛОМ;
- зростання потреби користувачів у наданні послуг по обміну рухомими і нерухомими зображеннями.

У розвитку обчислювальних мереж спостерігаються дві тенденції:

- з одного боку, існує тенденція об'єднання локальних мереж (LAN) у міські (MAN) і глобальні (WAN) мережі з можливістю забезпечення високошвидкісного обміну;

- з іншого боку, у зв'язку зі швидким ростом продуктивності робочих станцій і ПЕОМ, а також у зв'язку з тим, що станції стають мультимедіа-терміналами, існує тенденція різкого підвищення швидкості роботи в самих локальних мережах.

## 20.6. Комутація локальних мереж..

Проблема комутації локальних мереж виникла як проблема ефективного використання пропускної здатності каналу у випадках високого та середнього завантаження мережі та як проблема сполучення окремих локальних мереж в одну мережу, в тому числі і глобальну.

Мережа, що використовує технологію комутації ЛМ, побудована за топологією розподіленої зірки. У вузлах зірки розташовані комутатори (Рис.20.2 ).

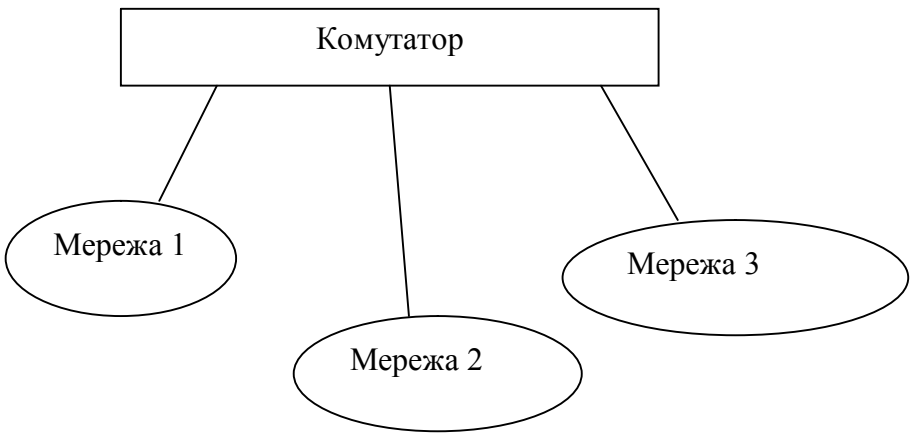



Рис.20.2

Вони,як і інші пристрої сполучення (мости, маршрутизатори, шлюзи), аналізують та фільтрують інформаційний потік, у результаті чого частина інформаційного потоку локалізується в одній частині мережі і не завантажує іншої частини мережі.. Кожен комутатор має порти, до яких приєднані окремі станції або цілі сегменти мережі. Комутація не тільки підвищує перепускну здатність, а й забезпечує більший захист інформації в мережі = інформація передається у відповідний сегмент, а не у всю мережу.

Зіркова топологія та наявність комутаторів роблять мережу стійкою до збоїв, надійною в роботі.

Близько розташовані комутатори (концентратори) можно сполучати в конфігурацію, що має назву *стек.* Комутаторні стеки використовуютьтоді, коли виникає потреба збільшити кількість портів. Це найдешевший спосіб збільшити кількість приєднаних станцій та мереж. Комутатори бувають з буферизацією і без буферизації.. В складних мережах доцільно використовувати комутатори з буферизацією. Буферизація дозволяє з'єднувати порти з різними швидкостями передавання. В мережі в масштабах однієї робочої групи дешевше використовувати комутатор без буферизації.

Комутатори – це альтернатива концентраторам та маршрутизаторам. Системи на базі маршрутизаторів коштують значно дорожче і мають більшу затримку передавання внаслідок аналізу мережевих таблиць маршрутизації та виконання функцій мережевого рівня. Маршрутизатори доцільно використовувати у великих мережах з територіально виділеними офісами. В інших системах краще використовувати комутатори. Існує великий набір комутаторів з різним призначенням: комутатори мережі підприємств, комутатори відділів- для взаємодії робочих груп, комутатори, що забезпечують швидкий зв'язок одного або кількох портів і таке інше. Вибір комутатора- одна з ключових задач при проектуванні комп'ютерної мережі.

## 20.7. Віртуальні локальні мережі

З об'єднанням локальних мереж і з перетворенням корпоративних мереж у великі комплекси, що сполучають десятки ЛМ та тисячі ПК, виникла ідея створення з комп'ютерів, рознесених на великі відстані, сукупність комп'ютерівякі взаємодіяли б як станції однієї ЛМ – так званої *віртуальної* локальної мережі (ВЛМ). Віртуальній мережі повинна надаватись частина перепускної здатності всієї мережі. Об'єднання у віртуальну мержу комп'ютерів за певною ознакою повинно бути динамічним – входження і вихід ПК з ВЛМ повинно здійснюватись без значних зусиль адміністратора. Створення ВЛМ за певним типом сервісу вирішує крім того і задачу незалежності між фізичною та

логічною структурою мережі. Логічна мережа не повинна залежати від змін у фізичній структурі мережі.. Цим вимогам найкраще відповідають мережі АТМ, які надають певні сервіси за запитом.

При організації авіртуальних мереж у магістральному каналі мережі до службових повідомлень протоколу, який керує потоком, додається поле, що ідентифікує ВЛМ. Це можуть бути МАС-адреси, адреса локальної мережі, тип протоколу, а також комбінація цих ознак. НайпростішийвипадокВЛМ – це віртуальний сегмент. Довільну кількість сегментів об’єднують в один віртуальнийщо функціонує як замкнений домен трафіку. Після визначення такої мережі весь трафік між визначеними фізичними сегметами комутується зішвидкістю передавання фізичного середовища та з мінімвльними затримками. Для керування віртуальними мережами застосовують мультипротокольні маршрутизатори (Рис.20.3).

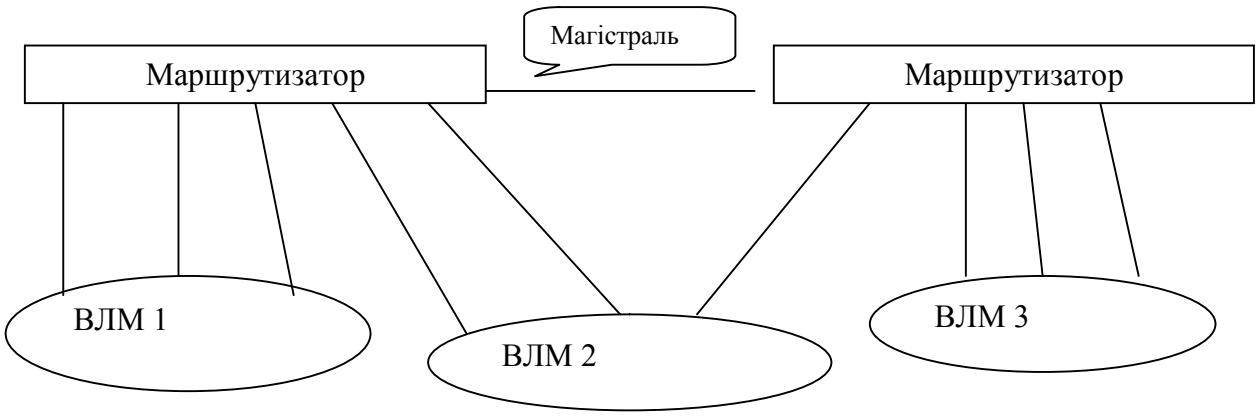



Рис.20.3

Існують віртуальні підмережі (тут працюють комутатори). Для мережі одного типу протоколів (TCP/IP) немає потреби в маршрутизаторах. Їх замінюють комутатори. У випадку ВЛМ з таблицею МАС-адрес для кожної ВЛМ створюється своя таблиця МАС-адрес.

## 20.8. Комутація третього рівня.

Комутатори працюють як багатопортові мости на канальному рівні. На мережевому рівні працюють маршрутизатори. Така схема добре працювала у межах робочих груп, де сервер знаходився в тому ж сегменті, що і користувачі. Тут виконувалось співвідношення внутрішніх і зовнішніх передавань як 80% до 20%. Згодом, в зв’язку із збільшенням мереж, з можливостями працювати з віддаленим сервером, з можливостями використовувати Web-простір пропорція в розподілі потоків докорінно змінилась. Це викликало необхідність використання дорогих маршрутизаторів для великих міжмережевих потоків, що стало невигідним та викликало велику затримку передавання у мережах. Тому виникла концепція комутації третього рівня, в якій замість маршрутизаторів використовують більш швидкі і дешевші комутатори.

Технології, які реалізують функції мережевого рівня у комутаторах, мають назву комутації третього рівня. Тут комутатор аналізує адреси і якщо кадр відноситься до однієї і тієї ж мережі, виконується комутація кадру. Інакше - здійснюється маршрутизація.

Для цього в комутаторах вмонтовано пристрій для маршрутизації.

## 20.9. Організація складних зв'язків у глобальних мережах з використанням мостів NETWARE.

У глобальних мережах зв'язок між ЛОМ здійснюється за допомогою мостів.

*Мости* — являють собою програмно-апаратні комплекси, що з'єднують ЛОМ між собою, а також ЛОМ і окремі робочі станції (РС), дозволяючи їм взаємодіяти один з одним для розширення можливостей збору й обміну інформацією.

Міст звичайно визначається як з'єднання між двома мережами, що використовують однаковий протокол взаємодії, однаковий тип середовища передачі й однакову структуру адресації.

Існує два базових типи мостів NETWARE:

- внутрішній;
- зовнішній.

Якщо міст розташовується у файловому сервері — *внутрішній міст*.

Якщо міст розташовується в робочій станції — *зовнішній міст*.

Зовнішні мости і їх ПЗ встановлюються в робочій станції, що функціонує не як файловий сервер. Тому зовнішній міст може передавати дані більш ефективно, чим внутрішній.

Існують виділені і сполучені мости.

*Виділений* — це комп'ютер, що використовується як міст, але не може функціонувати як робоча станція.

*Сполучений* — може функціонувати і як міст, і як робоча станція — одночасно.

*Перевага*: обмежуються витрати на покупку додаткового комп'ютера.

*Недолік*: обмеженість потенційних можливостей робочої станції, розміщеної в ньому.

Коли прикладна програма на РС зависає і викликає зупинку РС, що функціонує як міст, програма моста також зупиняє операції. Цей збій перериває поділ даних між мережами, а також перериває сеанси роботи РС, що зв'язані через міст із файловим сервером.

Оскільки виділений міст не використовується як РС, то ніякі ППП не викликають такий збій і не переревають роботу.

Вибираючи міст, необхідно враховувати вартість устаткування і ризик можливості збою моста.

Локальний міст передає дані між мережами, що розташовані в межах обмежень кабелю по відстані. Локальні мости застосовуються в наступних випадках:

1. — для поділу великих мереж на дві і більш підмереж з метою збільшення швидкодії і зменшення вартості ліній зв'язку.

Наприклад, в одній організації різні відділи розділяють одну й ту ж мережу. Великі мережі повільніше малих, тому доцільно виділити в них

невеликі підмережі компактно розташованих відділів. Використовуючи локальний міст Netware, відділи можуть продовжувати розділяти дані таким чином, як в одній мережі, здобуваючи при цьому швидкодію і гнучкість, властиві малій мережі.

2. — за допомогою локального моста можна розширити фізичні

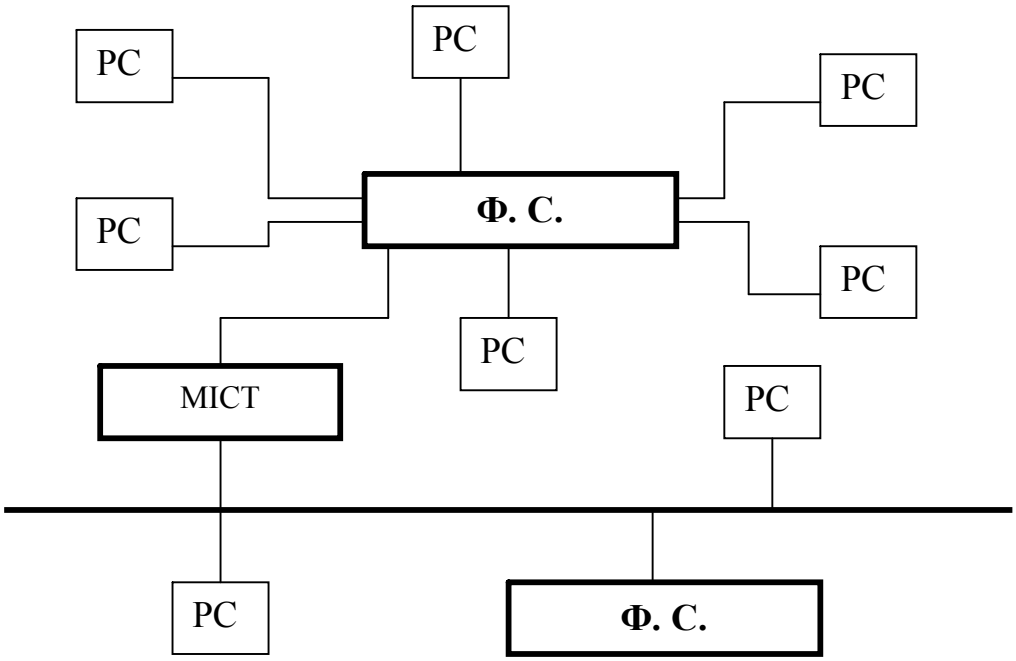



**Рис. 20. 4.** Розширення фізичних можливостей мереж

можливості мережі. Якщо мережа Netware має максимально припустиме число вузлів, підтримуване її апаратною схемою адресації і є необхідність у додаванні ще декількох вузлів, то для розширення такої мережі використовується міст Netware. При цьому включення в мережу додаткового файлового сервера (Ф.С.) необов'язково.

3. — об'єднання мереж в інтермережу.

Щоб користувачі кожної мережі могли отримувати доступ до інформації інших мереж, необхідно зв'язати ці мережі, утворити інтермережу.

Виділені мости застосовуються, коли відстань не дозволяє з'єднувати мережі за допомогою кабелю.

Наприклад: з'єднання мережі в м. Костромі з мережею м. Новгорода поставить перед необхідністю у використанні виділеного моста, тому що обмеження по довжині кабелю для локального моста буде перевищено.

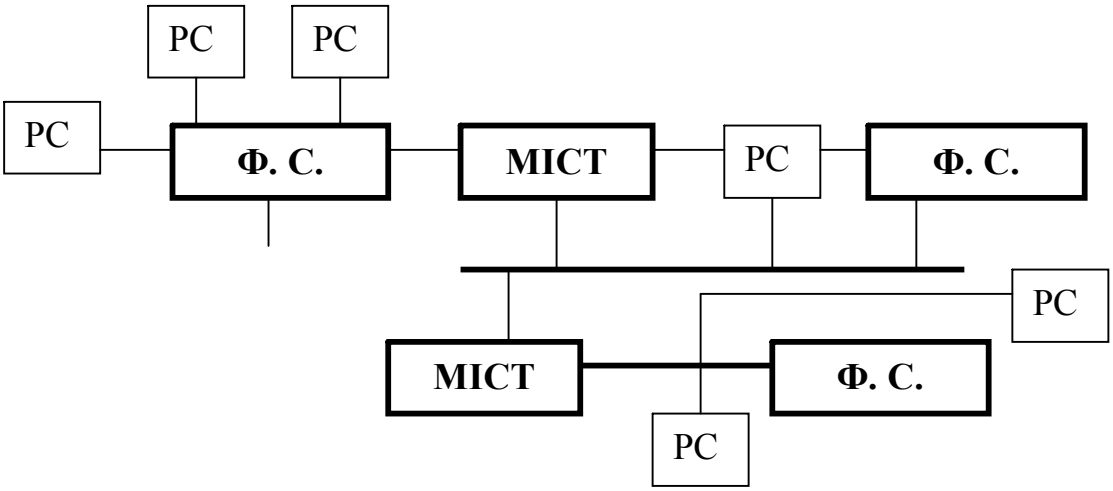



**Рис. 20. 5.** Приклад інтермережі

Виділений міст використовує проміжне середовище передавання (телефонні лінії) для з'єднання з віддаленою мережею чи віддаленими РС.

При з'єднуванні мережі з віддаленою мережею необхідно установити міст на кожному кінці з'єднання, а при зв'язку мережі з віддаленим РС — міст потрібно тільки на мережі.

Вибір модемів для організації віддаленої взаємодії повинний визначатися характеристиками і типом каналів зв'язку, а також вимогами до можливостей модемів і їхньої вартості.

*Примітка*:

- до 2400 бод — телефонні канали зв'язку (1бод=1біт/сек при частотній та амплитудній модуляціях), використовуються з низько і середньо — швидкісними асинхронними модемами (асинхронний);
- до 19,2 Кбод — у виділених лініях,(синхронний); (звичайна телефонна лінія, що має максимальну швидкість V=64 Кбіт/с, або комутована телефонна лінія з швидкістю передавання даних V=9600 біт/с).

Виділені мости Netware підтримують два види методів послідовної передачі: асинхронний і синхронний.

Основне розходження між мостом у захищеному (protected — mode) режимі і мостом у реальному (real — mode) режимі полягає в обсягу пам'яті, що він може підтримувати.

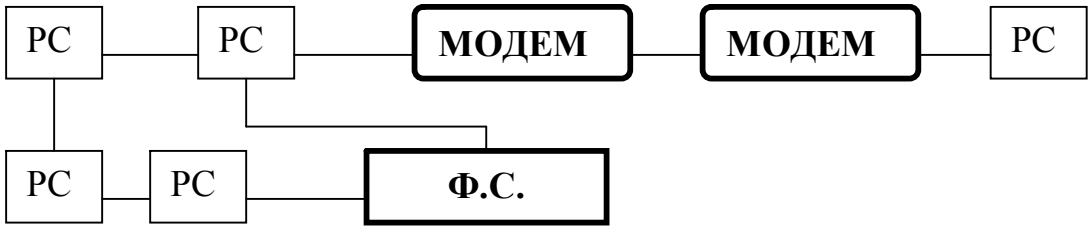



**Рис. 20. 6.** Віддалене з’єднання з використанням зовнішнього мосту

Захищений міст дозволяє додавати пам'ять, у той час як реальний міст надає мінімум пам'яті.

*Міст у захищеному режимі*. ПЗ моста в захищеному режимі підтримує стандартний 1 Мбайт пам'яті моста (640 Кб ОЗУ + додаткова пам'ять). Воно (ПЗ) також підтримує установку плат пам'яті в загальному обсязі до 8 Мб. Цей обсяг додаткової пам'яті дозволяє мати міст, на якому можуть виконуватися додаткові процеси (Value Added Processes — VAP) в обсязі пам'яті аж до 7 Мб.

Якщо планується установити більш ніж один або два VAP — процеси, варто вибрати міст у захищеному режимі. При цьому необхідно визначити додаткову кількість плат пам'яті. Число плат, що доповнюються, залежить від того, скільки VAP-процесів планується виконувати. Якщо буде виконуватися більш ніж два VAP-процеси, необхідно установити принаймні одну плату.

*Примітка*. Якщо потрібно виконувати 4 VAP-процеси, наприклад таких, як VAP друк і VAP обслуговування черги, міст повинен працювати в захищеному режимі.

Перш ніж використовувати міст у захищеному режимі, необхідно переконатися у можливості типу комп'ютера для роботи в сполученому режимі.

*Міст у реальному режимі*. ПЗ моста в реальному режимі підтримує стандартні 640 Кб основної пам'яті, у цьому випадку в мосту може

виконуватися один чи два додаткових орієнтованих процеси (VAP). Мости в реальному режимі можуть бути як виділеними, так і сполученими.

Обчислювальна мережа дозволяє користувачам мережі використовувати у своїх роботах сервіс мережного друку. Мережними друкувальними пристроями (ПД) можуть бути принтери, плоттери чи будь-які периферійні пристрої.

ПД є мережним, якщо він підключений ззовні до робочої станції (РС) чи мережі, і може бути використаний в інтересах різних користувачів чи груп користувачів мережі з різних ділянок мережі. Останні моделі сучасних ПД мають великі функціональні можливості, високу продуктивність. Вони досить дороги і застосування їх у виді локальних буде сполучено з великими матеріальними витратами.

Сервіс друку NETWARE дозволяє відразу декільком користувачам більш ефективно його використовувати. Наприклад, один лазерний принтер фірми XEROX, підключений у мережу, дасть можливість заощадити засоби, не здобуваючи інші.

Коли немережева станція надсилає запит на друк на підключений до неї принтер, цей запит відразу ж спрямовується на виконання. Якщо користувач буде працювати з мережними принтерами, то інформація, що він виводить на ПД, буде спрямована спочатку у файловий принтер чи - сервер, а вже потім на принтер. Коли принтер готовий виконувати черговий запит, принтер-сервер вибирає завдання на друк з черги і посилає його на принтер, що відповідає даній черзі. Принтер-сервер є складовою частиною програмного компонента файл-сервера, що вибирає завдання на друк з черги і направляє їх у принтер. Принт-сервер може також бути присутнім у мережі у вигляді спеціалізованої робочої станції, що покликана обслуговувати процес друку в мережі, або він може бути сполучений з ПЗ моста. У мережі процес мережного друку може здійснюватися і на принтерах, підключених до звичайної окремої РС.

Принтер-сервер NETWARE збільшує можливості друку мережі, він може обслуговувати до 16 принтерів, під’єднаних до різних комп'ютерів мережі, і

може бути інстальованим (інсталяція — установка програмного забезпечення на ПЕОМ) на файл-сервері, чи на мосту спеціалізованої РС.

ПЗ принтер-сервера звичайно сполучено з ПЗ файлового сервера і використовує VAP-процеси, що завантажуються на файл-сервері. VAP-процеси принтер-сервера використовують у процесі роботи на сервері чи мосту до 128 Кб пам'яті, включаючи DOS при завантаженні на мосту. Для кожного принтера додається ще по 10 Кб.

При використанні спеціалізованого принтер-сервера на РС для його роботи потрібно 200 Кб пам'яті, плюс по 10 Кб для кожного підключеного принтера. Ці цифри можуть мінятися в залежності від завантаженості принт-сервера.

Окремий принтер вимагає для своєї РС 9 Кб пам'яті. Ця цифра включає й обсяг буфера, необхідний для роботи принтера. Окремий принтер буде функціонувати при відключеному файловому сервері, якщо принт-сервер оформлено у вигляді спеціалізованої РС чи інстальовано на мосту.

У системі NETWARE процес друку реалізований у такий спосіб: оболонкою РС направляє файл по мережі у файловий чи принтер -сервер, де він, відповідно до системного планування, буферизується і ставиться в чергу з параметрами завдання для друку.

При одночасному надсиланні інформації користувачами на друк, запит, отриманий першим, буде оброблений у першу чергу. Усі наступні запити вбудовуються в чергу і будуть оброблені в цій же послідовності, якщо тільки вони не одержать вищий пріорітет.

Робочим завданням на друк слугують характеристики, що визначають, як повинен виконуватисся друк. До них відносяться: режим, формат, кількість копій, а також вказівка на конкретний принтер, що буде виконувати роботу. Кожен користувач створює завдання на друк і відсилає його в файл- чи принтер-сервер, де воно вже ставиться в чергу.

NETWARE версії 2.15 дозволяє одному принтеру обслуговувати кілька черг, і одна черга може обслуговуватися декількома принтерами. Наприклад,

при наявності декількох запитів на друк, принтерам Printer0 і Printer1 може бути дане завдання на виконання черги з більш високим пріорітетом.

Можна також визначити, яким користувачам дозволено розміщувати

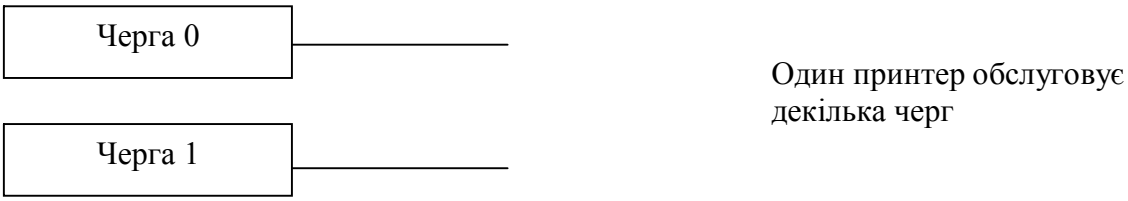



**Рис. 20. 7.** Організація черг

завдання на друк в кожну чергу.

Будь-яка черга на друк повинна бути спланована за допомогою спеціальних засобів. Можна установити відповідність між чергами на друк і принтерами за допомогою команд, що вводяться з консолі файл-сервера, чи з підготовленого файлу autoexec.sys.

## 20. 10. Питання для самоконтролю

8. У залежності від способу організації обробки даних і взаємодії користувачів які виділяють два типи інформаційних систем?
9. Що виконує центральний процесор в ієрархічних мережах?
10. Переваги та недоліки ієрархічних систем?
11. Що таке клієнт і сервер у системах клієнт-сервер?
12. Які прийнято виділяти за організацією взамоєдії типи систем, що використовують метод клієнт-сервер?
13. Що таке рівноправна мережа? Переваги та недоліки.
14. Що таке мережа з виділеним сервером? Переваги та недоліки.
15. Які три методи передачі даних використовуються при обміні даними між вузлами?
16. Асинхронна передача даними. Переваги та недоліки.
17. Синхронна передача даними. Переваги та недоліки.
18. Типи кабелів, що використовуються в ЛОМ?
19. Що таке скручена пара?

20. Екранований та неекранований кабель.
21. Коаксіальний кабель.
22. Оптоволоконний кабель.
23. Безпровідні локальні мережі. Переваги та недоліки.
24. Комунікаційна мережа. Переваги та недоліки.
25. Мережна технологія ATM.
26. Що таке віртуальне з'єднання?
27. Що таке віртуальний канал?
28. Як діє ATM технологія?
29. Які новітні технології створюються на базі ATM технології?
30. Які функції забезпечуються новітніми технологіями ШЦМІО?

## 21. ТЕХНОЛОГІЯ ВІДКРИТИХ СИСТЕМ

В наш час актуальним являється перехід від невеликих локальних мереж персональних комп'ютерів до промислових корпоративних інформаційних систем — UPSIZING. Більшість середніх і великих державних і комерційних організацій поступово відмовляються від використання тільки ПК, задача сьогоднішнього дня — створення відкритих і розподілених інформаційних систем.

На сьогоднішній день розвиток інформаційних технологій — створення єдиних мереж підприємств і корпорацій, що поєднують віддалені комп'ютери і локальні мережі, що часто використовують різні платформи, у єдину інформаційну систему. Таким чином необхідно об'єднати користувачів комп'ютерів у єдиний інформаційний простір і надати їм спільний доступ до ресурсів. Однак тут виникає безліч труднощів, зв'язаних з рішенням задачі по організації каналів зв'язку (кабель Ethernet не прокладеш по місту, а тим більше до іншого кінця планети). При побудові корпоративних мереж іноді використовуються телефонні канали, але зв'язок по таких лініях, що комутуються, ненадійний, оренда виділених ліній зв'язку дорога, а ефективність такого каналу невисока. Проблема виникає і при інтегруванні в корпоративну

мережу різнорідних ЛОМ, а також у підключенні великих комп'ютерів, наприклад, IBM mainframe чи VAX. Складності виникають і при об'єднанні в одну локальну мережу комп'ютерів з різними ОС. Тому побудова корпоративної мережі задача не з легких.

*Проблема перша* — це канали зв'язку. Самим оптимальним варіантом є використання вже існуючих глобальних мереж передачі даних загального користування, щоб комунікаційний протокол у корпоративній мережі збігався з прийнятим в існуючих глобальних мережах. Найбільш раціональним вибором тут варто вважати протокол X.25. Даний протокол дозволяє працювати навіть на низькоякісних лініях зв'язку, тому що розроблявся він для підключення віддалених терміналів до великих ЕОМ і відповідно містить у собі могутні засоби корекції помилок, звільняючи від цієї роботи користувача.

Подальший розвиток X.25 — Frame Relay, а також нові протоколи типу ATM хоча й обіцяють великі швидкості, вимагають практично ідеальних ліній зв'язку і, можливо, не швидко будуть широко застосовуватися в найближчому майбутньому.

Протокол X.25 дозволяє організувати в одній лінії до 4096 віртуальних каналів зв'язку. Якщо прокластити до офісу одну виділену лінію. то її можна використовувати для об'єднання кількох окремих офісів, підключення корпоративних інформаційних ресурсів, доступу до систем електронної пошти, базам даним — одночасно.

Виділена лінія — це звичайна телефонна лінія, з якою можна працювати на швидкостях 9600-28800 біт/с. Більш швидкісні лінії (64 Кбіт/с і >) коштують значно дорожче.

Звичайно мережі X.25 будуються на двох типах устаткування — Switch чи центр комутації пакетів (ЦКП) і PAD (hfcket assembler/disassembler — збирач/розроблювач пакетів), названий також пакетним адаптером даних (ПАД), чи термінальним концентратором. ПАД служить для підключення до мережі X.25 кінцевих пристроїв через порти. Прикладом використання ПАД у

корпоративній мережі — підключення банкоматів до центрального комп'ютера банку.

*ЦКП* — його завдання полягає у визначенні маршруту, тобто у виборі фізичних ліній і віртуальних каналів у них, по яких буде пересилатися інформація.

Перехід до багатокористувацьких СУБД — якісно технологічний стрибок, що забезпечує діяльність організацій у майбутньому. Реалізація переходу до нової інформаційної системи (ІС) залежить від використовуваної і перспективної моделі клієнт-сервер.

*Моделі клієнт-сервер* — це технологія взаємодії комп'ютерів у мережі. Кожний з комп'ютерів має своє призначення і виконує свою визначену роль. Одні комп'ютери в мережі володіють і розпоряджаються інформаційно-обчислювальними ресурсами (процесори, файлова система, поштова служба, служба друку, база даних), інші мають можливість звертатися до цих служб, користаючись їхніми послугами.

Комп'ютер, керуючий тим чи іншим ресурсом називають сервером цього ресурсу, а комп'ютер, що користується ним — клієнтом.

Кожен конкретний сервер визначається видом того ресурсу, яким він володіє. Наприклад, призначенням сервера баз даних є обслуговування запитів клієнтів, зв'язаних з обробкою даних; файловий сервер, чи файл-сервер, розпоряджається файловою системою і т.ін.

Цей принцип поширюється і на взаємодію програм. Програма, що виконує надання відповідного набору послуг, розглядається як сервер, а програми що використовуються цими послугами, прийнято називати клієнтами. Програми мають розподілений характер, тобто одна частина функцій прикладної програми реалізується в програмі-клієнті, а інша — у програмі-сервері, а для їхньої взаємодії визначається деякий протокол.

Розглянемо ці функції. Один з основних принципів технології клієнт-сервер полягає в поділі функцій стандартного інтерактивного додатка на чотири групи, що мають різну природу.

*Перша група*. Це функції введення і відображення даних.

*Друга група* — поєднує чисто прикладні функції, характерні для даної предметної області (для банківської системи — відкриття рахунка, переклад грошей з одного рахунка на інший і т.ін.).

*Третя група* — фундаментальні функції збереження і керування інформаційно-обчислювальними ресурсами (базами даних, файловими системами і т.ін.).

*Четверта група* — службові функції, що здійснюють зв'язок між функціями перших трьох груп.

Відповідно до цього в будь-якому додатку виділяються наступні логічні компоненти:

- компонент представлення (presentation), що реалізує функції першої групи;
- прикладний компонент (business application), що підтримує функції другої групи;
- компонент доступу до інформаційних ресурсів (resource manager), що підтримує функції третьої групи, а також вводяться й уточнюються угоди про способи їхньої взаємодії (протокол взаємодії).

Розходження в реалізації технології *клієнт-сервер* визначаються наступними факторами:

- видами програмного забезпечення, у які інтегрований кожний з цих компонентів;
- механізмами програмного забезпечення, використовуваними для реалізації функцій усіх трьох груп;
- способом розподілу логічних компонентів між комп'ютерами в мережі;
- механізмами, використовуваними для зв'язку компонентів між собою.

Виділяються чотири підходи, реалізовані в наступних моделях:

1. модель файлового сервера (File Server — FS);
2. модель доступу до виділених даних (Remote Data Access — RDA);
3. модель сервера баз даних (Data Base Server — DBS);
4. модель сервера додатків (Application Server — AS).

## 21.1. Файловий сервер

*Модель файлового сервера* (FS) – є базовою для локальних мереж ПК. Донедавна була популярна серед вітчизняних розробників, що використовували такі системи, як FoxPro, Clipper, Clarion, Paradox і т.ін.

Одним з комп'ютерів у мережі вважається файловим сервером і надає іншим комп'ютерам послуги по обробці файлів. Файловий сервер працює під керуванням мережної операційної системи (Novell NetWare) і відіграє роль компонента доступу до інформаційних ресурсів (тобто до файлів). На інших ПК у мережі функціонує додаток, у кодах якого сполучені компонент представлення і прикладний компонент

Протокол обміну являє собою набір викликів, що забезпечують додатку доступ до файлової системи на *файл-сервері*.

До недоліків технології даної моделі відносять низький мережний трафік (передача безлічі файлів, необхідних додатку), невелику кількість операцій маніпуляції з даними (файлами), відсутність адекватних засобів безпеки доступу до даних ( захист тільки на рівні файлової системи) і т.ін.

## 21. 2. Доступ до виділених даних

Модель доступу до виділеним даних (RDA) — істотно відрізняється від FS-моделі методом доступу до інформаційних ресурсів. У RDA-моделі коди компонента представлення і прикладного компонента сполучені і виконуються на комп'ютері-клієнті. Доступ до інформаційних ресурсів забезпечується операторами спеціальної мови (SQL, якщо мова йде про бази даних) чи викликами функцій спеціальної бібліотеки (якщо мається спеціальний інтерфейс прикладного програмування — API).

Запити до інформаційних ресурсів спрямовуються по мережі виділеному комп'ютеру, що обробляє і виконує їх, повертаючи клієнтові блоки даних.

Говорячи про архітектуру клієнт-сервер, мають на увазі дану модель. Основна перевага RDA-моделі полягає в уніфікації інтерфейсу клієнт-сервер у

вигляді мови SQL і широкому виборі засобів розробки додатків. До недоліків можна віднести істотне завантаження мережі при взаємодії клієнта і сервера за допомогою SQL-запитів; неможливість адміністрування додатків у RDA, тому що в одній програмі сполучаються різні за своєю природою функції (представлення і прикладні).

## 21. 3. Сервер баз даних

*Модель сервера баз даних* (DBS) - реалізована в деяких реляционных СУБД (Informix, Ingres, Sybase, Oracle).

Її основу складає механізм збережених процедур — засіб програмування SQL-сервера. Процедури зберігаються в словнику баз даних, розділяються між декількома клієнтами і виконуються на тому ж комп'ютері, де функціонує SQL-сервер. У DBS-моделі компонент представлення виконується на комп'ютері-клієнті, у той час як, прикладний компонент, оформлений як набір збережених процедур, функціонує на комп'ютері-сервері БД. Там же виконується компонент доступу до даних, тобто ядро СУБД.

Поняття інформаційного ресурсу звужено до баз даних, оскільки механізм збережених процедур — відмінна характеристика DBS-моделі — мається поки тільки в СУБД.

*Переваги* DBS-моделі:

- можливість централізованого адміністрування прикладних функцій;
- зниження трафіку (замість SQL-запитів по мережі направляються виклики збережених процедур);
- можливість поділу процедури між кількома додатками;
- економія ресурсів комп'ютера за рахунок використання одноразово створеного плану виконання процедури.

*До недоліків* відноситься:

- обмеженість засобів написання збережених процедур, що представляють собою різноманітні процедурні розширення SQL, що поступаються образотворчими засобами та функціональними можливостями перед такими

мовами як Pascal. Сфера їх використана обмежена конкретною СУБД через відсутність можливості налагодження і тестування різноманітних збережених процедур.

На практиці частіше використовуються змішані моделі, коли цілісність бази даних і деякі найпростіші прикладні функції забезпечуються збереженими процедурами (DBS-модель), а більш складні функції реалізуються безпосередньо в прикладній програмі, що виконується на комп'ютері-клієнті (RDA-модель).

## 21.4. Сервер додатків

*Модель сервера додатків* (AS) - являє собою процес, виконуваний на комп'ютері-клієнті, відповідальний за інтерфейс із користувачем (тобто реалізує функції першої групи).

Прикладний компонент реалізований як група процесів, що виконують прикладні функції, називається сервером додатка (Application Server — AS).

Доступ до інформаційних ресурсів здійснює менеджер ресурсів (наприклад, SQL-сервер). З прикладних компонентів доступні такі ресурси як, бази даних, черги, поштові служби й ін. AS, розміщена на комп'ютері, де функціонує менеджер ресурсів, рятує від необхідності застосування SQL-запитів по мережі, що підвищує продуктивність системи.

Моделі RDA і DBS спираються на дволанкову схему поділу функцій:

- у RDA-моделі прикладні функції надані програмі-клієнту (прикладний компонент зливається з компонентом представлення);
- у DBS-моделі відповідальність за їхнє виконання бере на себе ядро СУБД (прикладний компонент інтегрується в компонент доступу до інформаційних ресурсів).

У AS-моделі реалізована трьохланкова схема поділу функцій. Тут прикладний компонент виділений як найважливіший ізольований елемент додатка. Порівнюючи моделі, AS має більшу гнучкість і має універсальний характер.

*Принципи переходу до нової інформаційної системи.*

При переході до нової інформаційної системи (ІС) необхідно вирішити такі питання як вибір однієї з чотирьох моделей, компоненти архітектури ІС і інструментарій переходу.

Найбільш розповсюдженої ІС є FS-модель (приймемо її за вихідну), а в якості цільовий — RDA-модель (найбільш поширена і відносно проста). На практиці спостерігаються й інші схеми переходу ( FS—>DBS, RDA— > DBS, RDA—>AS, FS—>AS). Найбільш типовий випадок це FS—>RDA, це перехід від локальних мереж ПК до архітектури систем із сервером баз даних.

Наступний крок — визначення компонентів архітектури системи, що має у своїй основі RDA-модель — комп'ютер-клієнт і сервер баз даних. Проблема полягає у виборі апаратного і базового програмного забезпечення цих компонентів.

На сьогоднішній день використовуються ПК на базі процесорів Pentium під керуванням ОС/2 та MS Windows (поширеність, популярність, велике число додатків, широкий набір активно використовуваних русифікованих продуктів). Найважливіша перевага MS Windows — безліч засобів швидкої розробки додатків, що працюють з SQL-орієнтованими СУБД і доступність цих засобів для вітчизняних користувачів.

Говорячи про сервер БД, необхідно згадати, що це повинен бути могутній комп'ютер, облаштований високошвидкісними надійними механізмами дискової пам'яті великої ємності і системою архівування на магнітних стрічках. Його робота повинна здійснюватися під керуванням багатозадачної багатокористувацької ОС, що підтримує промислові стандарти.

Для RDA-моделі характерні два ключових компоненти:

- ПК на базі процесорів 486/Pentium під керуванням ОС MS Windows;
- високопродуктивний RISC-комп'ютер (фірм Sun, Hewlett-Packard, IBM) під керуванням відповідної версії ОС UNIX.

## 21. 5. Питання для самоконтролю

1. Яка тенденція розвитку інформаційних технологій?
2. Що таке єдиний інформаційний простір?
3. Яка проблемо виникає при створені корпоративних мереж?
4. Проблема каналів зв'язку.
5. Що дозволяє протокол X.25?
6. Подальший розвиток протоколу X.25.
7. Що таке виділена лінія?
8. На яких типах устаткування будується мережа X.25?
9. Що означає перехід до багатокористувацьких СУБД?
10. Моделі *клієнт-сервер*.
11. Який комп'ютер називають сервером?
12. Яку програму називають сервером, а яку — клієнтом?
13. Треба розглянути один з основних принципів технології *клієнт-сервер* в поділі функцій стандартного інтерактивного додатка.
14. Які з функцій стандартного інтерактивного додатка групи, що мають різну природу?
15. Якими факторами визначаються розходження в реалізації технології *клієнт-сервер*?
16. Модель файлового сервера (FS).
17. Модель доступу до виділених даних (RDA).
18. Яким чином спрямовуються запити до інформаційних ресурсів?
19. Модель сервера баз даних (DBS).
20. Модель сервера додатків (AS).
21. Принципи переходу до нової інформаційної системи.

## 22. Технологія роботи в середовищі розподіленої обробки даних

Однієї з найважливіших мережних технологій є розподілена обробка даних, що дозволяє підвищити ефективність задоволення інформаційної потреби користувача і, забезпечити гнучкість і оперативність прийнятих їм рішень.

*Перевагою* розподіленої обробки інформації є:

- велике число взаємодіючих між собою користувачів;
- усунення пікових навантажень з централізованої бази даних за рахунок розподілу обробки і збереження локальних баз даних на різних ЕОМ;
- можливість доступу користувача до обчислювальних ресурсів мережі ЕОМ;
- забезпечення обміну даними між вилученими користувачами.

При розподіленій обробці виконується робота з базою, тобто представлення даних, їхня обробка, робота з базою на логічному рівні здійснюється на комп'ютері клієнта, а підтримка бази в актуальному стані — на сервері. При наявності розподіленої бази даних база розміщується на декількох серверах. На сьогодні створені бази даних в усіх напрямках людської діяльності: економічного, фінансового, кредитного, статистичного, науково-технічної, маркетингу, патентної інформації, електронної документації і т.ін.

Створення розподілених баз даних (РБД) було викликано двома тенденціями обробки даних, з одного боку — інтеграцією, а з іншого боку — децентралізацією.

Інтеграція має на увазі централізоване керування і ведення баз даних. Децентралізація забезпечує збереження даних у місцях їх виникнення і обробки, при цьому швидкість обробки підвищується, вартість знижується, збільшується ступінь надійності системи.

Розподілена база даних — база даних, частини якої розміщені на окремих ЕОМ, що входять у мережу. При цьому деякі дані можуть дублюватися.

При проектуванні РБД здійснюється розбивка об'єкта на кілька частин (фрагментів) і розміщення кожного фрагмента на один чи кілька комп'ютерів.

Розміщення фрагментів може бути надлишковим чи безнадмірною. При надлишковому розміщенні необхідно визначити ступінь дублювання фрагментів. Вигоди, одержувані від дублювання, пропорційні співвідношенню обсягів вибірки даних і їхнього відновлення. Для підтримки цілісності бази даних потрібно коректування всіх копій. Переваги дублювання зменшуються зі збільшенням вартості збереження фрагментів. Ефективність роботи користувачів із РБД залежить від забезпеченості їх інформацією про дані в РБД, їхній структурі і розміщенні. Цю задачу вирішує мережний словник-довідник даних, що знаходиться в одній ЕОМ чи в мережі (дублюється на декількох ЕОМ). При цьому, словник-довідник може мати розподілену структуру, тобто коли його окремі фрагменти розподілені по робочих станціях мережі.

До організації баз даних пред'являються такі загальні вимоги як, забезпечення високою швидкістю обробки запитів, таємності, незалежності (фізичної і логічної) даних, безпеки і т.д. Крім перерахованих вимог, до РБД висуваються вимоги «прозорості»: розподіленої структури БД; спільного доступу до даних; розподіленої обробки.

Розподілена структура БД припускає незалежність кінцевих користувачів і програм від способу розміщення інформації на робочих станціях мережі, тобто формулювання запитів до РБД виконується аналогічно запитам до централізованої БД.

Спільний доступ до даних означає можливість модифікації цих даних декількома користувачами не порушуючи цілісності РБД.

«Прозорість» розподіленої обробки означає незалежність користувачів і програм від типу локальної обчислювальної мережі і застосовуваного мережного програмного забезпечення. Обробка запиту користувача може виконуватися на декількох ЕОМ.

Доступ користувачів до РБД і адміністрування здійснюється за допомогою системи управління розподіленою базою даних (СУРБД), що забезпечує виконання наступних функцій:

- автоматичне визначення ЕОМ, що зберігає необхідні в запиті дані;

- декомпозицію розподілених запитів на часткові підзапити до БД окремого ЕОМ;
- планування обробки запитів;
- передавання часткових підзапитів і їхнє виконання на виділених ЕОМ;
- прийом результатів виконання часткових підзапитів;
- підтримка в погодженому стані копій дубльованих даних на різних ЕОМ мережі;
- керування рівнозначним доступом користувачів до РБД;
- забезпечення цілісності РБД.

## 22. 1. Питання для самоконтролю

1. Розподілена обробка даних.
2. Чим було викликано створення розподілених баз даних?
3. Що забезпечує інтеграція баз даних?
4. Що забезпечує децентралізація баз даних?
5. Розподілена база даних.
6. Ефективність роботи користувача в РБД.
7. Які вимоги висуваються при організації баз даних?
8. Що припускає розподілена структура БД?
9. Що означає спільний доступ до даних?
10. Які функції забезпечує система управління розподіленою базою даних?

## 23. Базові технології обробки запитів

Прикладні програми керування даними являють собою необхідний інструмент для розподіленої обробки.

Архітектура *клієнта-сервер* мережі дозволяє різним прикладним програмам одночасно використовувати загальну базу даних. Зовсім очевидно, що перенесення програм керування даними з робочих станцій на сервер сприяє вивільненню ресурсів робочих станцій, надає можливість збільшити число часток, локально розв'язуваних задач. Дана архітектура дозволяє також централізувати ряд найважливіших функцій керування даними, такі, як захист інформації баз даних, забезпечення цілісності даних, керування спільним використанням ресурсів.

Однією з важливих переваг архітектури *клієнт-сервер* в мережній обробці даних є можливість скорочення часу реалізації запиту. У підтвердження цьому розглянемо дві базові технології обробки інформації в архітектурі *клієнт-сервер* мережі і технології використання традиційного файлового сервера.

Припустимо, що прикладна програма бази даних завантажена на робочу станцію і користувачу необхідно одержати всі записи, що задовольняють деяким пошуковим умовам. У середовищі традиційного файлового сервера програма керування даними, що виконується на робочій станції, повинна здійснити запит до сервера кожного запису бази даних (рис.23.1, а). Програма керування даними на робочій станції може визначити, чи задовольняє запис пошуковим умовам, лише після того, як вона буде передана на робочу станцію.

Очевидно, що даний технологічний варіант обробки інформації має найбільший сумарний час передавання даних по каналах мережі.

У середовищі *клієнт-сервер*, напроти, робоча станція надсилає запит високого рівня серверу бази даних. Сервер бази даних здійснює пошук записів на диску й аналізує їх. Записи, що задовольняють умовам, можуть бути накопичені на сервері. Після того, як запит цілком оброблений, користувачу на

робочу станцію передаються всі записи, що задовольняють пошуковим умовам (рис. 23.1, б).

Дана технологія дозволяє знизити мережний трафік і підвищити пропускну здатність мережі. Більше того, за рахунок виконання операції доступу до диска й обробки даних в одній системі сервер може здійснити пошук і обробляти запити швидше, ніж якби ці запити оброблялися на робочій станції.

Прикладні програми баз даних *клієнт-сервера* підтримуються програмними продуктами:

- NetWare Btrieve- програмою керування записами з індексацією по ключу (виконується на сервері);
- NetWare SQL — ядром реляційних баз даних, призначеним для забезпечення системи захисту і цілісності даних.

Служби баз даних NetWare Btrieve і NetWare SQL фірми Novell дозволяють розробникам створювати надійні прикладні програми баз даних без необхідності написання власних програм керування записами, що забезпечує зручне перенесення прикладних програм у середовище *клієнт-сервер.*

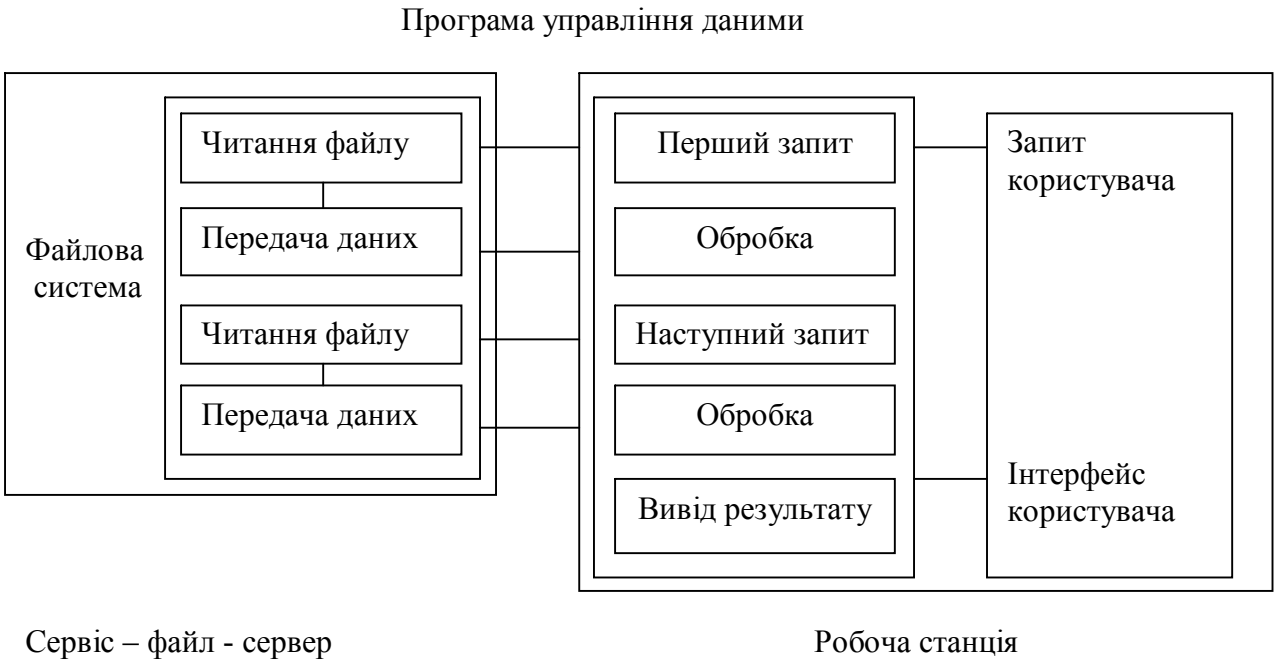
На сьогодні розроблені десятки тисяч прикладних автономних і багатозадачних програм, орієнтованих на клієнта версій NetWare Btrieve, NetWare SQL, що можуть бути використані організаціями, які створюють чи мають мережі ЕОМ. Більше того, версії NetWare Btrieve і NetWare SQL фірми Novell для клієнтів мають погоджені API, що спрощує перенесення програм із середовища одного клієнта в середовище іншого.

За ступенем мінливості всі бази даних (БД) можна розділити на два класи:

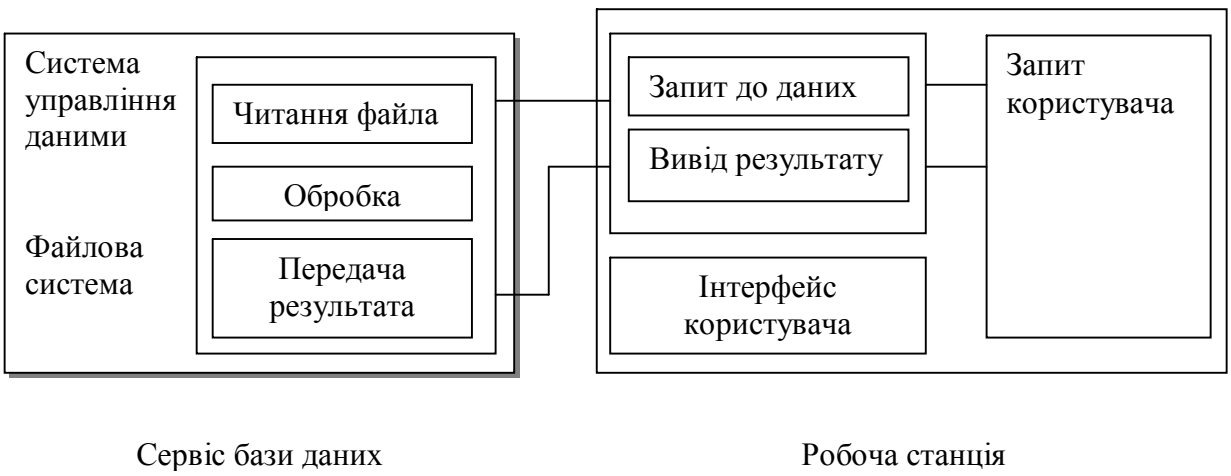
А — постійний-постійну-постійне-постійна-умовно-постійні (в основному для довідкових систем);

Б — сильно динамічні (для банківських, біржових систем і т.п.).

Для ведення баз даних першого і другого класів використовуються системи управління базами даних (СУБД), що у значній мірі відрізняються одна від одної як за функціональними можливостями, так і за експлуатаційними характеристиками.




а) Типове середовище обробки запитів у мережах ЕОМ.




б) Розподілене середовище обробки запитів у мережах ЕОМ.

**Рис. 23.1** (а,б).Технології обробки запитів по базових варіантах

Наприклад,

- для постійних-умовно-постійних БД найбільш важливими показниками є показники швидкості опрацьовування запитів і швидкості формування вихідних звітів по БД, а такі показники, як швидкість опрацьовування транзакцій і контроль цілісності БД при опрацьовуванні транзакцій не дуже критичні;
- для сильно динамічних БД на перший план виходять такі показники, як швидкість опрацьовування транзакцій, можливість контролю цілісності, швидкість формування звітів, погодженість по читанню і транзакціяж. Менш критичні тут швидкості опрацьовування запитів.

Тому будь-яка СУБД не може однаково успішно застосовуватися при роботі з БД різних класів. Такі системи, як CLIPPER, FOXPRO орієнтовані на перший клас БД-(А), і тут маються непогані результати, а такі СУБД ,як Informix, Ingres, SyBase створювалися для другого класу (Б).

Виходячи з вищеозначеного напрошується висновок: знайти «золоту середину», що задовольняла б вимогам обох класів (А) і (Б). Рішенням цієї суперечливої задачі є використання диференціальної організації файлів бази даних, чи диференціальних файлів (ДФ).

Останнім часом розробники СУБД провідних фірм підійшли до використання ідеї ДФ. Причинами цього явилися наступні факти:

- значно розширився клас розв'язуваних на IBM PC задач, так, що термін «персональний комп'ютер» уже не відповідає дійсності;
- широке поширення локальних обчислювальних систем (ЛОС);
- розробка багатокористувацьких і багатозадачних систем;
- стрімкий розвиток технічної бази ЕОМ (у більшому ступені дискової пам'яті).

Зупинимося на суті ДФ стосовно до БД у ЛОС. Реалізація ідеї ЛОС у різних СУБД значно відрізняється.

Ідея ДФ включає три положення:

- основний файл БД залишається незмінним при будь-яких відновленнях бази даних, тобто, будь-які зміни БД послідовно накопичуються в спеціальному файлі змін (не плутати з журналом транзакцій) — ДФ;
- ніякі індекси для нього не створюються і не підтримуються;
- коли ДФ досягне значних розмірів (приблизно 25-40% від розмірів БД), адміністратор вносить усі зміни в основний файл БД у зручний момент часу в пакетному режимі.

Як приклад візьмемо порівняння книги, де спостерігаються помилки в сторінках, і бази даних із ДФ. Немає необхідності перевидання книги через декілька помилок чи незначних змін. Якщо ця кількість має тенденцію до значного росту і досягло граничного значення, то стають виправданими витрати на перевидання книги, куди повинні бути включені всі накопичені зміни.

Переваги ДФ відносяться до забезпечення високої надійності, цілісності БД і швидкості опрацьовування транзакцій.

Питання, які швидкості опрацьовування транзакцій можна забезпечити при використанні ДФ, є досить важливим. Очевидно, що швидкість опрацьовування транзакций при такій організації БД зросте в десятки разів. При цьому сервер бази даних практично нагадує звичайний файл-сервер.

Що стосується індексів, то проблеми їхньої підтримки не існує (швидкості додавання, видалення і модифікації записів БД знаходяться на найвищому рівні). Внесення додавань у БД не відрізняється від додавань у звичайний послідовний файл. Час відновлення записів БД не залежить ні від розмірів БД, ні від довжини ключів, ні від їхньої кількості. Тимчасові витрати на блокування (як одне з вузьких місць для БД і ЛОС) зведені до мінімуму.

Для того, щоб забезпечити погодженість даних по читанню немає необхідності блокувати цілком таблицю, що має місце в ряді СУБД, тобто, коли запит (чи формований звіт) починає виконуватися, СУБД «запам'ятовує» старшу адресу в ДФ (моментальний знімок). При цьому користувач, що викликає свій запит, не зобов'язаний чекати «свого моменту». Він «не бачить»

нікого з користувачів і одержує знімок БД саме в цей момент часу. Далі, у міру виконання запиту (навіть дуже швидкого) частина цілей-записів-мет могла бути змінена чи вилучена. Це відіб'ється тільки на старших адресах ДФ, а тому СУБД просто проігнорує будь-які зміни даних, що випадково з’явились після початку виконання запиту. Гарантується коректування складних і тривалих запитів до БД, тобто забезпечення погодженості по читанню і транзакціях.

Постає цікаве питання, як у цьому випадку ведеться пошук у БД. У цьому випадку по ассоциатору знаходиться безліч цілей-записів-мет: число і список їхніх адрес в основну БД, після чого здійснюється зчитування «ассоциатора» ДФ і виконується коректування цього списку. За рахунок цього коректування час пошуку збільшується, причому величина цього збільшення залежить від розмірів ДФ. Своєчасність відновлення БД повинна бути в компетенції адміністратора БД. Щоб виключити істотні витрати, зв'язані з ДФ, можна накопичувати зміни БД для їхньої пакетної обробки і при пошуку ДФ не враховувати. У ряді систем, таких як банківські, допускається втрата деякої точності в період між циклами відновлення — «контрольоване запізнювання».

Крім цього використання ДФ забезпечує:

- можливість адміністратору відновлювати випадково вилучені записи;
- можливість (при необхідності) зберігати індексні файли на самих робочих станціях;
- можливість створення розподілених БД;
- одночасне виконання транзакцій.

Несуперечність даних може забезпечуватися механізмом захоплення на рівні запису — відкочування транзакцій будь-якої доступної вкладеності.

## 23. 1. Питання для самоконтролю

1. Що собою являють прикладні програми керування даними?
2. Чи дозволяє архітектура *клієнт-сервера* мережі різним прикладним програмам одночасно використовувати загальну базу даних?
3. Чому сприяє перенесення програм керування даними з робочих станцій на сервер?
4. Яка перевага архітектури *клієнт-сервера* в мережній обробці даних?
5. Якими програмними продуктами підтримуються прикладні програми баз даних *клієнт-сервера*?
6. На які два класи можна розділити по ступені мінливості всі бази даних (БД)?
7. Чи може будь-яка СУБД однаково успішно застосовуватися при роботі з БД різних класів?
8. Використання диференціальної організації файлів бази даних.
9. Які факти з'явилися причинами використання ідеї диференціальних файлів?
10. Суть диференціальних файлів стосовно до БД у ЛОС.
11. Які положення включає ідея диференціальних файлів?
12. Переваги диференціальних файлів.
13. Що забезпечує диференціальні файли?

## 23. 2. Контрольні роботи за темами 20, 21, 22, 23

| Варіант | Питання параграфа 20.10. | Питання параграфа 21.5. | Питання параграфа 22.1 | Питання параграфа 23.1 |
|---|---|---|---|---|
| 1 | 1, 16 | 1, 16 | 1 | 1 |
| 2 | 2, 17 | 2, 17 | 2 | 2 |
| 3 | 3, 18 | 3, 18 | 3 | 3 |
| 4 | 4, 19 | 4, 19 | 4 | 4 |
| 5 | 5, 25 | 5, 21 | 5 | 5 |
| 6 | 6, 26 | 6, 20 | 6 | 6 |
| 7 | 7, 27 | 7, 19 | 7 | 7 |
| 8 | 8, 28 | 8, 18 | 8 | 8 |
| 9 | 9, 29 | 9, 17 | 9 | 9 |
| 10 | 10, 30 | 10, 16 | 10 | 10 |
| 11 | 11, 20 | 1, 15 | 1 | 11 |
| 12 | 12, 21 | 2, 14 | 2 | 12 |
| 13 | 13, 22 | 3, 13 | 3 | 13 |
| 14 | 14, 23 | 4, 12 | 4 | 1 |
| 15 | 15, 24 | 5, 11 | 5 | 2 |

## ЛІТЕРАТУРА


1. **Богумирский Б.** Энциклопедия Windows 98. – СПб: Питер Ком, 1999. – 816 с.

2. **Ботт, Эд, Леонард, Вуди.** Использование Microsoft Office 2000. Специальное издание: Пер.с англ.: Уч. пос. – М.: Издательский дом «Вильямс», 2000. – 1024 с.

3. **Гарнаев А. Ю.** Excel, VBA, Internet в экономике и финансах. — СПб.: БХВ-Петербург, 2001. — 816 с.

4. **Дибкова Л.М.** Інформатика та комп'ютерна техніка: Посібник для студентів вищих навчальних закладів. — К.: Видавничий центр «Академія», 2002. — 320 с.

5. **Дорот В.,** Новиков Ф. Толковый словарь современной компьютерной лексики. — СПб.: БХВ-СПб., 1999.

6. **Інформатика** для юристов и экономистов / Симонович С.В. и др. — СПб.: Питер, 2001. — 688 с.

7. **Інформатика:** Комп'ютерна техніка. Комп'ютерні технології. Посіб. / За ред. О.І.Пушкаря.— К.: Видавничий центр «Академія», 2001. — 696 с. (Альма-матер)

8. **Інформаційні** системи і технології в економіці: Посібник для студентів вищих навчальних закладів / За редакцією В. С. Пономаренка. — К,: Видавничий центр «Академія», 2002.— 544 с.

9. **Коваленко М.М.** Комп'ютерні віруси і захист інформації. Навч. Посіб. . — К.: Наук. Думка, 1999.

10. **Корнеев В. В., Гареев А. Ф., Васютин С. В., Райх В. В.** Базы данных. Интеллектуальная обработка информации. — М.: «Нолидж», 2000. — 352 с.

11. **Новиков Ф.А., Яценко А.Д.** Microsoft Office XP в целом. — СПб.: БХВ-Петербург, 2002. — 928 с.

12. **Основы** современных компьютерных технологий: Учеб. пособие.Под ред. А.Д. Хомоненко. — СПб.: ООО «Корона», 1998.

13. **ОУГЛТРИ ТЕРРИ** Microsoft Office XP в целом. — СПб.:ООО «ДиаСофтЮП», 2002. — 848 с.

14. **Руденко В.Д., Макарчик О.М. Патланжоглу М.О.** Практичний курс інформатики.—К.: Фенікс, 1997.— 304 с.

15. **Свириденко С. С.** Современные информационные технологии.— М.: Радио и связь, 1989. — 304 с.

16. **Фигурнов В.Э.** IBM PC для пользователя.— М.: Финансы и статистика, 2000.— 284 с.

17. **Хєлворсон М., Янг М.** Эффективная работа с Microsoft Office 2000. — СПб.: «Питер», 2001. — 1232 с.

18. **Хєлворсон М., Янг М.** Эффективная работа с Microsoft Office 97. — СПб.: «Питер», 2000. — 1056 с.

19. **Буров Є.**Комп'ютерні мережі.-Львов.: Бак, 1999.-408 с.

20. **Олифер В.**Г.,**Олифер Н.А**. Компъютерные сети: Принципы, технологии, протоколы .- СПБ.: "Питер", 2000.-672 с.

21. **Дуглас Э. Крамер.** Компъютерные сети и Internet.Разработка приложений для Internet.. Перевод с англ..-М.: "Вильямс", 2002, 640 с.