Офісне програмування

Об’єктна модель та програмування в Microsoft Excel

Слайди до лекцій
(4 частина)

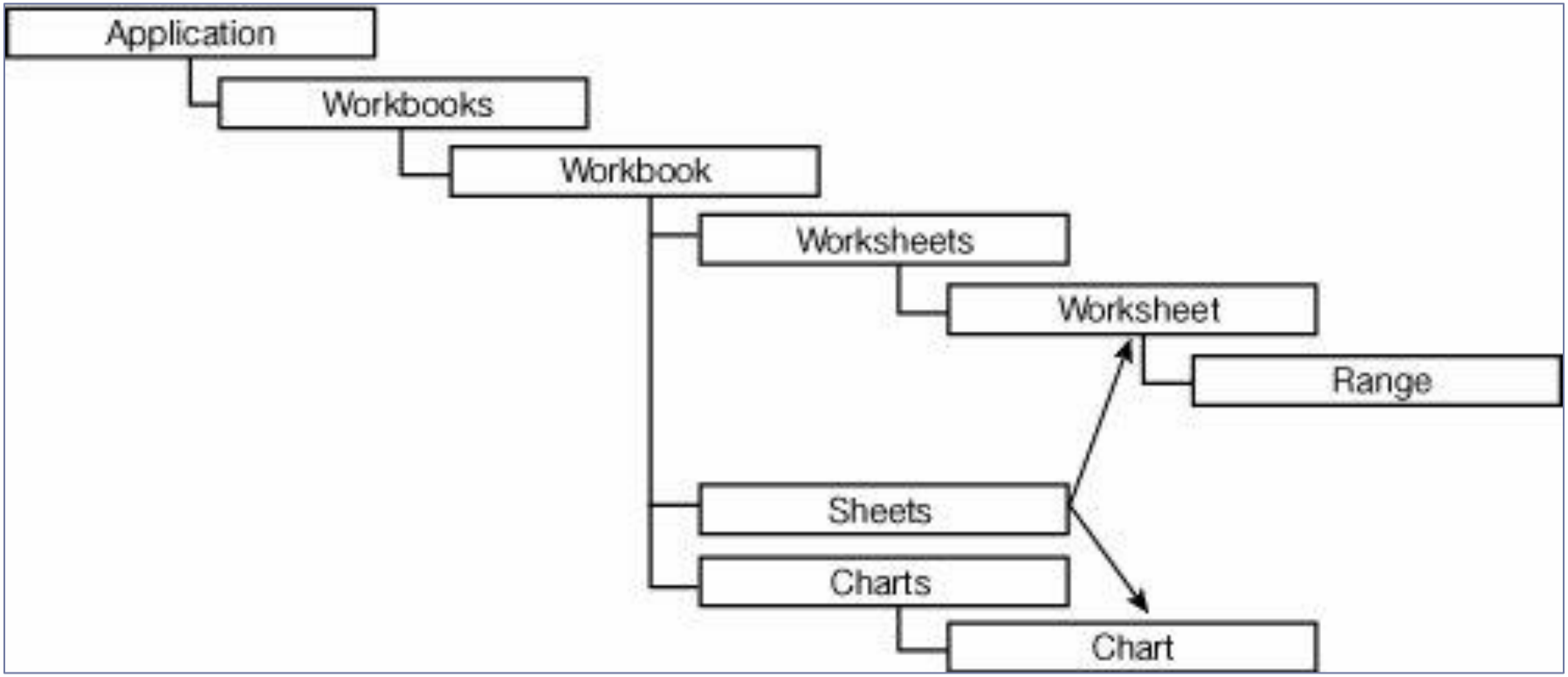
Приклади програм в Excel

- Програми, які виконують регулярні розрахунки, наприклад, фінансове планування, складання кошторисів і т.і;
- Автоматизація завантаження даних в таблицю Excel з бази даних, обробку цієї таблиці (розрахунки, моделювання тощо), і представити цю інформацію в стандартному вигляді;
- Генерація звітів у форматі файлів Excel;
- Завантаження даних з файлів Excel до бази даних;
- …

Відмінності застосування від Word

- Програмування в Excel найчастіше використовується не як засіб для введення і редагування даних, а для виконання різних розрахунків і відображення їх в спеціальних форматах (графік, зведена таблиця і т.і.).
- Навігація по книгам і листам Excel набагато зручніша, оскільки кожна комірка має свою адресу, окрім того, в Excel існує можливість присвоювати імена діапазонам комірок.

Об'єктна модель MS Excel



В Excel вбудовано велику кількість функцій (статистичних, фінансових, математичних і т.і.), які можливо використовувати у програмах

Об’єктна модель Excel містить більше 200 класів

https://docs.microsoft.com/en-us/previous-versions/office/developer/office-xp/aa141044(v=office.10)?redirectedfrom=MSDN

Об'єкт Application

- об'єкт Application в Microsoft Excel являє собою додаток Excel. Створити цей об'єкт - означає запустити Excel на комп'ютері.

```
Dim oExcel As New Excel.Application
```

за замовчуванням Excel запускається в прихованому вікні;

```
Dim oExcel As New Excel.Application
Dim oWbk As Workbook
oExcel.Visible = True
Set oWbk = oExcel.Workbooks.Add()
Debug.Print oExcel.Name; " - "; oWbk.Name
oWbk.Close
oExcel.Quit
```

Sub StartExcel()

Об'єкт Application

Якщо Word вже відкритий, то можна отримати на нього посилання:

```
Set oExcel = GetObject(, "Excel.Application")
```

- функція `GetObject([pathname], [class])` повертає посилання на об’єкт, вказаний аргументом `class` у форматі `appname.objecttype`, де `appname` - назва застосунку, який надає об’єкт, `objecttype` - тип або клас об’єкту, на який функція повертає посилання.
- Аргумент `pathname` містить повний шлях та ім’я файлу, який містить об’єкт, якщо цей аргумент не вказується, аргумент `class` є обов’язковим.

Sub GetExcel()

Особиста книга макросів - PERSONAL.XLSB

- Окрім створених користувачем робочих книг в Excel автоматично відкривається так звана *Особиста книга макросів* - PERSONAL.XLSB.
- *Особиста книга макросів* - PERSONAL. XLSB призначена для зберігання у ній макросів, які повинні бути доступні для запуску для кожної користувацької робочої книги.
- *Особиста книга макросів* - PERSONAL.XLSB зберігається у форматі *Двоичная книга Excel* у каталозі C:\Users\ім'я_користувача\AppData\Roaming\Microsoft\Excel\XLSTART.

GetOpenWorkbooks()

Об'єкт Application

- Якщо код VBA виконується в Excel (тобто Excel вже запущений), то об'єкт `Application` створювати вже не потрібно. Він буде автоматично доступний.
- Якщо не вказано, до якого об'єкту відноситься та чи інша властивість або метод, компілятор VBA в Excel автоматично вважає, що це властивість або метод належить об'єкту `Application`. Тому такий код функціонально однаковий:

```
Application.Workbooks.Add
```

та

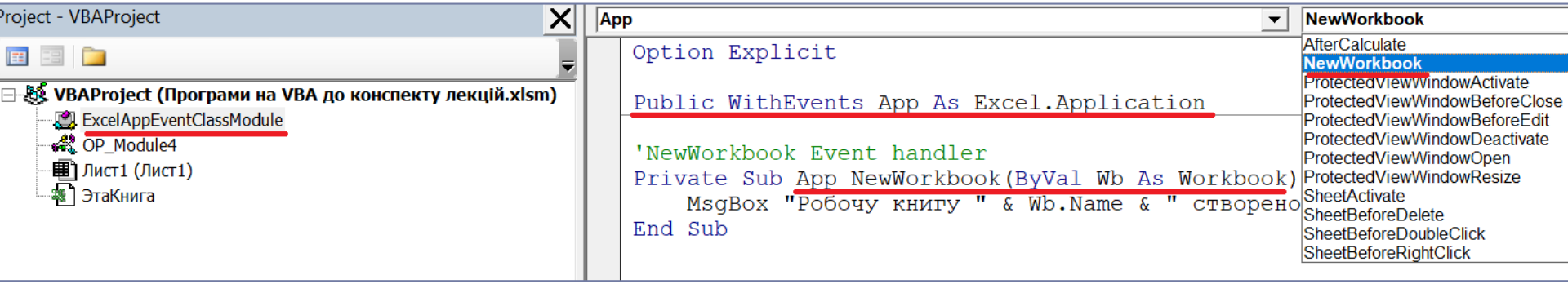
```
Workbooks.Add
```

Об'єкт Application - обробка подій об'єкта

Для того, щоб у вікні редактора коду для форм з'явився об'єкт `Excel.Application`, необхідно в розділі *Declarations* об’єктного модуля (модуля класу або форми) оголосити об'єкт `Excel.Application` з ключовим словом `WithEvents`, наприклад, так:

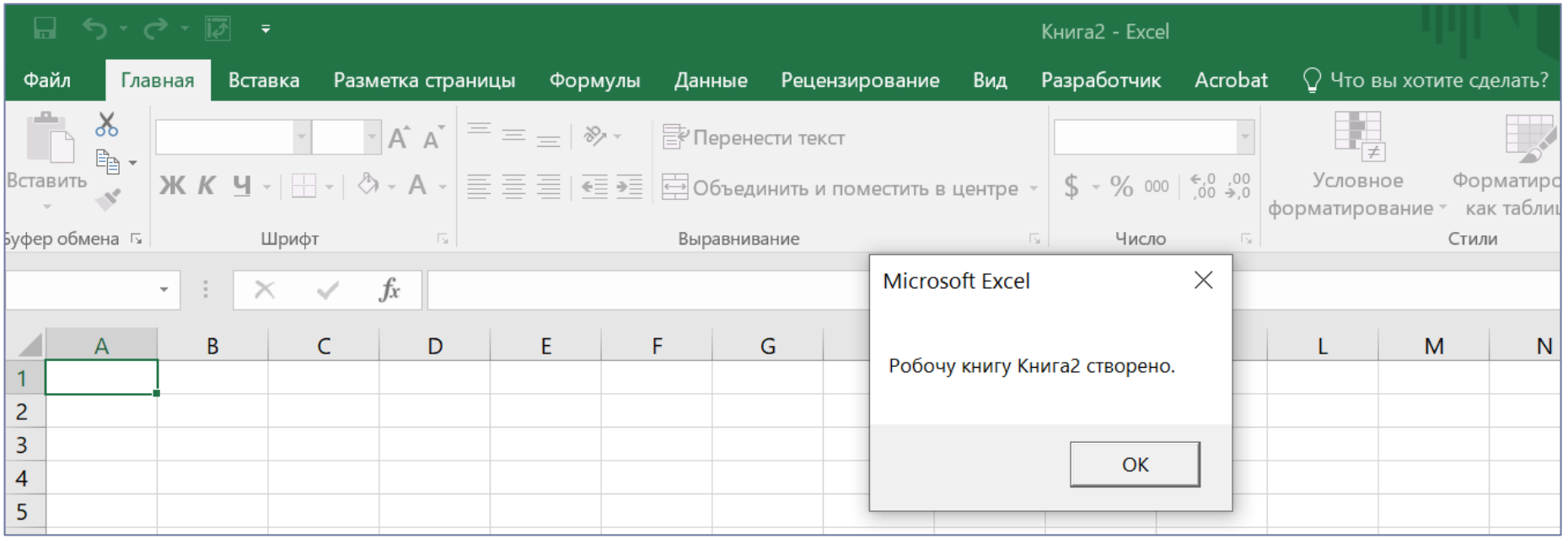
```
Public WithEvents App As Excel.Application
```

У списку об'єктів з'явиться новий об'єкт `App` (тобто `Application`), для якого можна вибрати події і додавати код в обробники подій.



Об'єкт Application

Тепер у звичайному модулі можна використовувати об’єкт класу, у якому оголошений об’єкт `Excel.Application` з подіями:



Виконання цього коду приведе до створення нової робочої книги додатком `eventApp.App` після чого автоматично буде запущений обробник події, який виведе інформаційне повідомлення

Sub CreateWorkbookWithEvent ()

Властивості об'єкта Application

- `Active…` - повертають відповідно активну комірку (ту, на яку вказує курсор введення даних), активний лист, активну книгу, активне вікно, активну діаграму. Властивості доступні тільки для читання, оскільки об'єкти `ActiveCell`, `ActiveSheet`, `ActiveWorkbook` і т.і. створюються автоматично під час роботи програми.
- `ActivePrinter` - дозволяє отримати або налаштувати активний принтер в ході роботи програми.
- `AddIns` - повертає однойменну колекцію надбудов (об'єктів `AddIn`). Разом з Excel поставляється кілька дуже корисних надбудов (вони доступні через меню *Файл-Параметры-Надстройки*), наприклад, *Пакет анализа*, *Поиск решения* і т.і. За допомогою цієї колекції можна перевіряти, чи підключена користувачем необхідна Вашій програмі надбудова і у разі неактивності підключити її.
- `AutoRecover` - повертає однойменний об'єкт, який дозволяє визначити параметри автозбереження Excel: каталог (`Path`) та час (`Time`). Наприклад, щоб відкриті документи Excel автоматично зберігались кожні 5 хвилин, можна використовувати код (можливо задавати час у діапазоні від 1 до 120 хвилин):

```
Application.AutoRecover.Time = 5
```

Властивості об'єкта Application

- `Calculation` - повертає або встановлює режим перерахунку робочої книги - автоматичний (за замовчуванням) - `xlCalculationAutomatic` = –`4105`, ручний - `xlCalculationManual` = –`4135` або напівавтоматичний - `xlCalculationSemiautomatic` = `2` (коли автоматично перераховується все, окрім таблиць). Є сенс відключати автоматичний перерахунок, коли він займає багато часу після зміни значень комірок.
- `CalculationState` - дозволяє перевіряти, чи виконується перерахунок - `xlCalculating` = `1` або він вже завершений `xlDone` = `0` або якщо розрахунок виконано, але перерахунок після змін - ще ні `xlPending` = `2`.
- `Cells` - одна з найважливіших властивостей - повертає об'єкт `Range`, який представляє усі комірки активного аркушу активної робочої книги. Оскільки для об'єкта `Range` властивість за замовчуванням - це властивість-об’єкт `Item`, то необхідні комірки активного листа зазначаються властивостями `RowIndex` і `ColumnIndex` цього об’єкту (їх значення - номери рядків та стовпців)::

```
Application.Cells(1,2).Font.Bold = True
```

рядок стовпець

Властивості об'єкта Application

- Дуже схоже діють властивості `Columns` і `Rows`. Наприклад, щоб виконати аналогічну операцію з цілим стовпчиком (другим), можна використовувати такий оператор:

  ```
  Application.Columns(2).Font.Bold = True
  ```

  а для рядка (другого) можна скористатися оператором:

  ```
  Application.Rows(2).Font.Bold = True
  ```

  Наголосимо, що властивості `Cells`, `Columns` і `Rows` містять посилання не на колекції об'єктів `Cell`, `Column` і `Row`, а колекції об'єктів `Range`. На використанні об'єкта `Range` побудована майже вся робота з комірками і їх значеннями.
- `Cursor` - дозволяє змінювати зовнішній вигляд курсора миші в Excel (у об'єкта `Word.Application` цієї властивості немає). Зазвичай перед виконанням довготривалого розрахунку курсору призначають вигляд пісочного годинника (`xlWait`), а після розрахунку - повертають звичайний вигляд (`xlDefault`). Після завершення роботи макросу курсор не повертається до звичайного вигляду автоматично, тому у програмі потрібно передбачити відповідний код.

Властивості об'єкта Application

- `DataEntryMode` - дозволяє перейти в режим введення даних - *data entry mode*, коли користувачеві дозволяється вводити дані тільки в розблоковані комірки обраного діапазону. Приймає значення: `xlOn` - включити цей режим, `xlStrict` - включити цей режим і не дозволити вийти з нього натисканням користувачем клавіші *<Esc>*, `xlOff` - відключити цей режим.
- `DecimalSeparator` і `ThousandsSeparator` - повертають та дозволяють призначити символ-роздільник цілої та дробової частини числа та символ-роздільник тисяч. При використанні цих властивостей рекомендується також відключити використання системних налаштувань за допомогою властивості `UseSystemSeparators`:

Sub ApplicationProperties()

```
Application.UseSystemSeparators = False
```

- `Dialogs` - повертає посилання на однойменну колекцію об’єктів `Dialog`, яку можна використовувати для відображення діалогових вікон Excel. Часто використовується об'єкт `FileDialog`, що представляє вікна, призначені для роботи з файлами (наприклад, вікно відкриття файлу).

Властивості об'єкта Application

- `FileDialog` - діалогове вікно відкриття і збереження файлів. Наприклад, щоб надати користувачеві можливість обрати єдиний файл у вікні відкриття (і отримати повний шлях до нього), можна скористатися кодом:

```
Application.FileDialog(msoFileDialogOpen) _
                              .AllowMultiSelect = False
Application.FileDialog(msoFileDialogOpen).Show
Debug.Print
Application.FileDialog(msoFileDialogOpen) _
                                    .SelectedItems(1)
```

Sub UseFileDialogOpen()

- `DisplayAlerts` - дозволяє відключати показ попереджень, які користувачеві при роботі програми показувати немає потреби.
- `EnableEvents` - дозволяє відключити події для об'єкта Application, так що вони генеруватись не будуть.
- `ErrorCheckingOptions` - містить посилання на однойменний об'єкт, за допомогою властивостей якого можна налаштувати параметри автоперевірки Excel (чи повідомляти користувачеві про синтаксично невірні вирази, посиланнях на порожні комірки і т.і.).

Властивості об'єкта Application

- `Interactive` - дозволяє заблокувати введення даних у Excel з боку користувача (як з клавіатури, так і за допомогою миші). Зазвичай використовується, щоб користувач не зміг перешкодити роботі програмі, наприклад, виконати якесь виділення. Можна також використовувати, якщо дані вводяться з іншої програми, яка взаємодіє з Excel.
- `International` і `LanguageSettings` - працюють аналогічно цим властивостям в Word.
- `LibraryPath` - містить шлях до каталогу надбудов Excel (файлів з розширенням .XLA), за замовчуванням - C:\Program Files (x86)\Microsoft Office\Office16\LIBRARY.
- `MoveAfterReturn` - дозволяє визначити, чи включений (`True`), або включити/відключити (`True/False`) перехід на наступну комірку після завершення введення даних і натискання на *<Enter>* (за замовчуванням - включений), а властивість `MoveAfterReturnDirection` дозволяє визначити напрямок переходу, зберігається як негативне ціле число: `xlDown = -4121, xlToLeft = -4159, xlToRight = -4161, xlUp = -4162`.

Властивості об'єкта Application

- `Names` - містить посилання на колекцію `Names`, що представляє всі іменовані `діапазони` в активній робочій книзі. За допомогою методу `Add()` колекції Names можливо визначати в робочій книзі іменовані діапазони, наприклад:

```
Names.Add Name:="test", _
                        RefersTo:="=Лист1!$a$1:$a$2"
```

  Іменовані діапазони використовують аналогічно закладкам Word - за їх допомогою зручно визначати набори даних в складних таблицях Excel. Іменовані діапазони в Excel можна визначити за допомогою команд секції *Определенные имена* меню *Формулы*.
- `ODBCErrors` і `OLEDBErrors` - дозволяють отримувати інформацію про помилки, що виникають при підключенні до баз даних ODBC і OLEDB, відповідно.
- `OnWindow` - це властивість більше схожа на подію. Як її значення вказується ім'я процедури, яка повинна знаходитися в модулі рівня книги (за замовчуванням такий модуль не створюється - його потрібно створити вручну). Ця процедура буде викликатися щоразу, коли користувач переходить у вікно Excel (не важливо з якою робочою книгою та її листом).

Властивості об'єкта Application

- `Range` - повертає об'єкт `Range`, який являє собою діапазон комірок і використовується в Excel для більшості операцій.
- `ReferenceStyle` - можливість переключити режим відображення адрес комірок між типом `xlA1` (літери - стовпці, цифри - рядки) і `xlR1C1` (коли і рядки, і стовпці вказуються цифрами). Користувачам ближче стиль виду `xlA1`, а програмістам - `xlR1C1` (особливо у тих ситуаціях, коли стовпців дуже багато і доводиться використовувати стовпці `AA`, `AB` і т.і.).
- `Selection` - повертає те, що в даний момент обрав/виділив користувач. Якщо він обрав діапазон комірок в таблиці, то повернеться об'єкт `Range`. Якщо ж користувач вибрав щось на діаграмі, то може повернутися об'єкт вісі, легенди і т.і. - в залежності від того, що було обрано. На відміну від Word, об'єкт `Selection` в Excel практично не використовується.
- `Sheets` - містить посилання на колекцію `Sheets`, яка включає звичайні листи книги і листи з діаграмами. Властивість `Worksheets` містить посилання на колекцію, що складається тільки зі звичайних листів (без листів з діаграмами).

Властивості об'єкта Application

- `TemplatesPath` - властивість для читання, яка містить повний шлях до каталогу з шаблонами Excel (наприклад, для розміщення власного шаблону або відкриття шаблону з цього каталогу). За замовчуванням використовується каталог AppData\Roaming\Microsoft\Шаблоны в профілі користувача.
- `ThisWorkbook` - дозволяє звертатися до поточної робочої книги, цей об'єкт створюється в Excel автоматично.
- `Windows` і `Workbooks` - повертають всі відкриті вікна і робочі книги, відповідно.
- `WorkSheetFunction` - дозволяє використовувати в програмі функції Excel безпосередньо, без необхідності прописувати їх в будь-яку комірку, наприклад:

```
Set myRange = Worksheets("Test").Range("A2:A10")
Debug.Print "Мінімальне значення у диапазоні "; _
myRange.Address; " = ";
WorksheetFunction.Min(myRange)
```

Sub ApplicationProperties()

Найважливіші методи об'єкта Application

- `ActivateMicrosoftApp()` - спеціальний метод, який призначений для запуску і активації (або просто активації, якщо програму вже запущено) застосунків MS Office і деяких інших застосунків Microsoft. Приймає як аргумент ціле число, що вказує на застосунок: `xlMicrosoftWord = 1, xlMicrosoftPowerPoint = 2, xlMicrosoftMail = 3, xlMicrosoftAccess = 4, xlMicrosoftFoxPro = 5, xlMicrosoftProject = 6, xlMicrosoftSchedulePlus = 7`.
- `Calculate...` - дозволяють перерахувати значення в комірках. Метод `Calculate()` виконує звичайний перерахунок значень (найчастіше використовується при вимкнутому автоматичному перерахунку), метод `CalculateFull()` дозволяє перерахувати значення у всіх відкритих робочих книгах, метод `CalculateFullRebuild()` додатково до перерахування значень у всіх відкритих робочих книгах виконує перебудову формул (аналогічно занесенню всіх формул заново). Зупинити перерахунок можна за допомогою методу `CheckAbort()`.

Найважливіші методи об'єкта Application

- `ConvertFormula()` - надає можливість перетворити формулу двома способами: або перевести адресацію комірок у інший режим (наприклад, замість `xlA1` в `xlR1C1`), або змінити абсолютні посилання на відносні і навпаки. Як параметр приймає строкову змінну з текстом формули (повинна починатися з символу =) і константи конвертації.
- `Evaluate(Name)` - дозволяє за ім'ям-рядком (довжиною < 256 символів) знайти об'єкт робочої книги Excel і перетворити його в інший об'єкт або значення для подальшого використання. Як параметр `Name` цей метод може приймати:
  - формулу, записану як рядок, якщо формула містить змінні-параметри, вони повинні бути рядками і додаватись через `&`
  - адресу однієї комірки в стилі `xlA1`, задану як рядок (повертається об'єкт `Cell`)
  - адресу діапазону, задану як рядок (повертається об'єкт `Range`)
  - імена, визначені у макросі (найчастіше - імена змінних) - повинні бути рядками
  - посилання на зовнішні робочі книги (наприклад, `Evaluate("[Book1.xls]Лист1!A5")`)

Найважливіші методи об'єкта Application

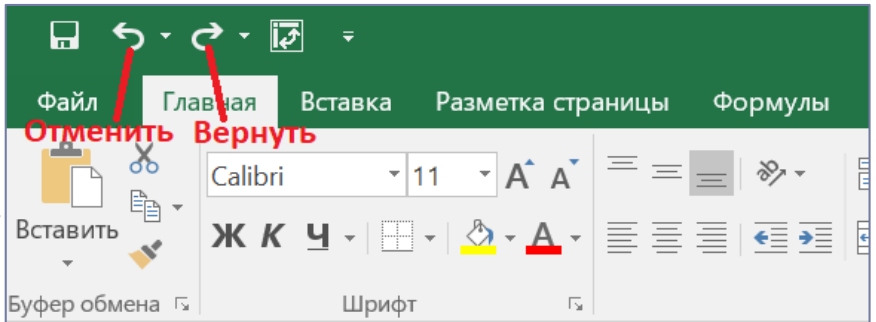
- об’єкти діаграм Ви можете вказати будь-яке ім'я об'єкта діаграми, наприклад "Название диаграммы" або "Легенда", щоб отримати доступ до властивостей та методів цього об'єкта

Функцію `Evaluate(Name)` можна викликати і неявно - вказавши ім'я об'єкта в квадратних дужках.

- `GetOpenFilename()` - найпростіший спосіб відкриття діалогового вікна *Open* і отримання інформації про вибір користувача (у вигляді строкової змінної з інформацією про ім'я файлу з повним шляхом)
- `GetSaveAsFilename()` - реалізує спосіб роботи з вікном *Save As*.
- `GoTo(Reference, Scroll)` - важливий і часто використовуваний метод - приймає як параметр `Reference` об'єкт `Range`, рядок з посиланням на комірку (в форматі `xlR1C1`) або ім'я-рядок процедури VBA і виділяє і активізує діапазон/комірку або запускає на виконання (якщо передається ім'я процедури). Значення `True` параметру `Scroll` виконує прокрутку листа так, щоб комірка, вказана параметром Reference відобразилась у верхньому лівому куті вікна. На відміну від методу `Select()`, цей метод ще й автоматично активізує лист, на якому розташований діапазон або комірка. Значення `False` параметру `Scroll` не виконує прокрутку листа (є значенням за замовчуванням).

Найважливіші методи об'єкта Application

- `Help()` - дозволяє відкрити інформацію з файлу довідки (у тому числі призначеного для користувача). Як параметр приймає ім'я файлу довідки і мітку теми в ньому.
- `Intersect()` - повертає діапазон (`Range`), який представляє прямокутну область перетину двох інших (або більше) діапазонів. Якщо діапазони, які передаються як параметри, не перетинаються, метод повертає `Nothing`.
- `OnKey()` - дозволяє задати процедурі VBA певну клавіатурну комбінацію. Для клавіш *<Alt>, <Ctrl>* та *<Shift>* використовують символи `%`, `^` та `+`, відповідно.
- `OnRepeat()` - надає можливість призначити процедуру, яка буде виконуватися при використанні команди *Вернуть* з *Панели быстрого доступа*
- Призначення процедури команді *Отменить* з тієї ж панелі виконується за допомогою методу `OnUndo()`



Найважливіші методи об'єкта Application

- `Repeat()` - дозволяє повторити дію, яку користувачем (не програмою) було виконано останньою. Працює аналогічно команді *Вернуть* з *Панели быстрого доступа*.
- Скасувати останню дію користувача можна методом `Undo()`. Працює аналогічно команді *Отменить* з *Панели быстрого доступа*.
- `Run()` - дозволяє виконати: процедуру або функцію VBA, макрос Excel, процедуру або функцію в модулі XLL (і передати їй до 30 параметрів). Параметром методу може бути або рядок з ім'ям макросу, або об'єкт `Range`, що вказує, де знаходиться функція. Якщо параметр - рядок, він буде оцінюватися в контексті активного листа.
- `SendKeys()` - дозволяє емулювати натискання клавіш і передати їх в активне вікно програми. Зазвичай використовується, якщо потрібно щось продемонструвати користувачеві (в цьому випадку використовуються паузи) або потрібно щось зробити через меню (наприклад, не вдалося знайти, як це можна зробити через код). Наступний код відкриває меню *File* (надається комбінація *<Alt>+F*):

```
Application.SendKeys() "%{f}"
```

Для клавіш *<Ctrl>* та *<Shift>* використовують символи ^ та +, відповідно.

Найважливіші методи об'єкта Application

- `Union()` - дозволяє об'єднати два або більше діапазонів.
- `Volatile()` - повинен викликатися тільки з користувацької функції, яка обчислює значення комірки. Якщо він викликаний і йому передано значення `True`, то ця комірка стає *чутливою* (*volatile*) і вона буде перераховуватися при будь-якій зміні значень в листі, навіть тих, які її не стосується.
- `Wait()` - дозволяє призупинити роботу Excel на зазначений як параметр час. Використовується для демонстрацій - щоб користувач встиг побачити, що відбувається, для очікування завершення виконання будь-якої зовнішньої операції і т.і. При цьому введення користувача блокується, а курсор миші набуває вигляду пісочного годинника. Однак деякі фонові операції Excel, такі, як друк і перерахунок значень, продовжують виконуватися.
- Методи `Quit(), OnTime(), FindFile()` працюють аналогічно однойменним методам в Word

Колекція Workbooks та об’єкти Workbook

- Об'єкт `Workbook` займає в Excel приблизно те ж місце, що і об'єкт `Document` в Word - він потрібен для отримання посилання на потрібну нам робочу книгу в наборі відкритих книг Excel.
- Отримати цей об'єкт можна кількома способами:
  - скористатися колекцією `Workbooks` об'єкта `Application` (вказувати `Application` не обов'язково - колекція Workbooks в Excel і так постійно доступна). Знайти потрібну книгу в цій колекції можна по її імені або номеру в колекції:

```
Debug.Print Workbooks("Успішність.xlsx") & _
                              .FullName
```

  - використовувати властивість `Application.ActiveWorkbook`.

```
Debug.Print ActiveWorkbook.Name
```

  - використовувати властивість `Application.ThisWorkbook`. При цьому ми звертаємося до тієї книги, яка належить поточному екземпляру Excel з програмним модулем, який виконується:

```
Debug.Print ThisWorkbook.Name
```

Колекція Workbooks та об’єкти Workbook

- На практиці частіше за все потрібно або створити в Excel нову книгу, або відкрити існуючу книгу. Для цієї мети використовуються методи `Add()` і `Open()`, відповідно:

```
Dim oWbk As Workbook
Set oWbk = Workbooks.Add()
```

Єдиний необов'язковий параметр, який приймає цей метод - повне ім'я існуючого файлу MS Excel, на основі якого створюється нова робоча книга (тобто створюється копія книги-параметру з новим ім’ям).

- Відкриття існуючої книги виглядає так:

```
Dim oWbk As Workbook
Set oWbk = WorkBooks.Open("C:\mybook1.xls")
```

- Крім стандартних, в колекції Workbooks передбачено також три спеціальних методи:

`OpenDatabase()` - відкрити базу даних, виконати до неї запит (або відкрити таблицю/уявлення безпосередньо), а результати запиту помістити як імпортовані зовнішні дані в нову автоматично створену робочу книгу Excel;

Колекція Workbooks та об’єкти Workbook

`OpenText()` - майже те ж саме, але в якості джерела тут виступає текстовий файл. Параметр методу `DataType` дозволяє визначити, чи використовується символ-роздільних даних у текстовому файлі (`xlDelimited`) або використовуються дані фіксованої ширини (`xlFixedWidth`). Параметри `Tab`, `Semicolon`, `Comma`, `Space`, `Other` типу `Boolean` дозволяють визначити, який саме символ-роздільник використовується. Наступний приклад відкриває текстовий файл з роздільником даних - комою. На відміну від методу `Open`, метод `OpenText()` не повертає посилання на відкритий об’єкт, тому отримати його можна таким чином:

```
Workbooks.OpenText Filename:=ActiveWorkbook.Path & _
Application.PathSeparator & "TESTFILE.txt", _
DataType:=xlDelimited, Comma:=True
Set owbkThird = ActiveWorkbook
```

`OpenXML()` - в якості джерела даних буде виступати файл у форматі XML. Як і метод `InsertDatabase()` в Word, цей метод слід використовувати тільки в найпростіших випадках.

Колекція Workbooks та об’єкти Workbook

- `SaveAs()` - дозволяє зберегти створену або відкриту робочу книгу. Обов’язковим параметром є `Filename` - повне ім’я файлу під яким буде збережена книга. Найбільш використовувані необов’язкові додаткові параметри: `FileFormat` - формат книги (доступний безпосередньо тільки для читання, можна змінювати при збереженні). Найбільш використовувані наведені в таблиці:

Ім’я	Значення	Опис	Розширення файлів
xlCSV	6	CSV	*.csv
xlExcel8	56	Excel 97-2003 Workbook	*.xls
xlHtml	44	HTML format	*.htm; *.html
xlOpenXMLTemplate	54	Open XML Template	*.xltx
xlOpenXMLTemplateMacroEnabled	53	Open XML Template Macro Enabled	*.xltm
xlOpenXMLWorkbook	51	Open XML Workbook	*.xlsx
xlOpenXMLWorkbookMacroEnabled	52	Open XML Workbook Macro Enabled	*.xlsm
xlTemplate	17	Excel Template format	*.xlt
xlWorkbookNormal	-4143	Workbook normal	*.xls

Колекція Workbooks та об’єкти Workbook

`ConflictResolution` - як будуть вирішуватися конфлікти зміни даних, якщо книга відкрита декількома користувачами відразу (shared workbook). Є можливість зробити так, щоб локальні зміни вносились автоматично (`xlLocalSessionChanges`), зміни віддалених користувачів вносились автоматично (`xlOtherSessionChanges`) або з’являлось діалогове вікно з можливістю визначити пріоритет змін (`xlUserResolution`). Однак для того, щоб не з’являлося діалогове вікно підтвердження перезапису файлу, необхідно відключити перед викликом методу `SaveAs()` виведення діалогових вікон, а після - включити:

```
Application.DisplayAlerts = False
owbkThird.SaveAs
Filename:="c:\Users\kgp\Dropbox\EDU\NewBook.xlsx", _
    ConflictResolution:=xlLocalSessionChanges
Application.DisplayAlerts = True
```

Sub AddOpenSaveWorkbook()

Найважливіші властивості об'єкта Workbook

- `Name` - ім'я файлу книги, `FullName` - ім'я файлу книги разом з повним шляхом до нього. `CodeName` - назва книги у проекті VBA (за замовчуванням - `ЭтаКнига`. Всі три властивості доступні тільки для читання, змінювати їх можна іншими способами (наприклад, зберігаючи файл під іншим ім'ям або безпосередньо у вікні *Properties*).
- Певне відношення до імен має також властивість `Path` (шлях до файлу книги).
- `Charts, Sheets, CustomViews, BuiltinDocumentProperties, CustomDocumentProperties, Windows, WebOptions` повертають однойменні колекції відповідних об'єктів.
- `ActiveSheet, ActiveChart` - властивості, що містять посилання на активний лист та активну діаграму робочої книги, відповідно.
- `Names` - повертає колекцію всіх іменованих діапазонів в даній робочій книзі. Цю властивість зручно використовувати для попередніх перевірок з метою усунення потенційних помилок часу виконання.

Sub WorkbookProperties()

Найважливіші властивості об'єкта Workbook

- `Name` - ім'я файлу книги, `FullName` - ім'я файлу книги разом з повним шляхом до нього. `CodeName` - назва книги у проекті VBA (за замовчуванням - `ЭтаКнига`. Всі три властивості доступні тільки для читання, змінювати їх можна іншими способами (наприклад, зберігаючи файл під іншим ім'ям або безпосередньо у вікні *Properties*).
- Певне відношення до імен має також властивість `Path` (шлях до файлу книги).
- `Charts, Sheets, Windows` повертають однойменні колекції відповідних об'єктів.
- `ActiveSheet, ActiveChart` - містять посилання на активний лист та активну діаграму робочої книги, відповідно.
- `Names` - повертає колекцію всіх іменованих діапазонів в даній робочій книзі.
- **Методів у об'єкта Workbook** також дуже багато, однак призначення найбільш вживаних - `Activate(), Close(), Save(), SaveAs(), PrintOut(), Protect() і Unprotect()` очевидне і діють воно аналогічно однойменним методам об'єкта Document в Word.

Sub WorkbookProperties()

Колекція Sheets (Worksheets) і об'єкт Worksheet

- В Word нижче об'єкта `Application` і `Document` починалися вже об'єкти безпосередньо для роботи з текстом - `Selection`, `Range` і т.і. В Excel між об'єктом робочої книги і комірками є ще один проміжний об'єкт - об'єкт `Worksheet` (*Лист*). Об'єкти `Worksheet` в книзі об'єднані в колекцію `Sheets` і `Worksheets`. Відміна між цими колекціями у тому, що елементами `Sheets` можуть бути не тільки робочі листи (об'єкти `Worksheet`), але й діаграми на окремих листах (об'єкти `Chart`).
- Найчастіше для введення даних в Excel необхідно в першу чергу визначитися з листом, куди будуть вводитися дані: для цього потрібно або вибрати його, або спочатку створити, а потім вибрати.
- Метод `Add()` колекції `Worksheets` приймає необов'язкові параметри: `Before` - містить посилання на лист, перед яким вставляється новий, `After` - містить посилання на лист, після якого вставляється новий, `Count` - кількість листів, які вставляються, `Type` - тип листа, що вставляється, може мати значення `xlWorksheet` - для звичайного листа, `xlChart` - для діаграми. Якщо нічого не вказувати, то новий лист буде розміщено найпершим.

Sub WorksheetCreation()

Колекція Sheets (Worksheets) і об'єкт Worksheet

- Для переходу до певного листа серед листів робочої книги використовується властивість `Item` колекції `Worksheets`, яка приймає як параметр індекс листа або назву листа, до якого потрібно перейти
- Для активації цього листа необхідно викликати з нього метод `Activate()`.
- Також колекції `Sheets` і `Worksheets` мають часто вживані властивості `Count` та `Item`, які дозволяють виконувати доступ до об’єктів `Worksheet`, і методи `Add()` та `Delete()`, які дозволяють додавати та видаляти листи робочої книги.
- У об’єкта `Worksheet` є часто вживані властивість `Visible` та методи: `Copy()`, `Move()`, `PrintOut()`, `PrintPreview()`, `Select()`, особливістю яких є те, що вони викликаються для об’єкта `Worksheet`, колекцій `Sheets` або `Worksheets`
- Також колекція `Worksheets` має метод `FillAcrossSheets(Range, Type)`, який дозволяє скопіювати об'єкт діапазону `Range` (варіанти: повністю (`xlFillWithAll=-4104`), тільки вміст (`xlFillWithContents=2`), тільки ознаки форматування (`xlFillWithFormats=-4122`)) до всіх робочих листів даної книги, діапазон повинен бути з робочого листа в колекції.

Sub WorksheetCreation()

Найважливіші властивості об’єкту Worksheet

- `Cells` - повертає об'єкт `Range`, який представляє всі комірки робочого листа. Працює аналогічно однойменній властивості `Application`, за винятком того, що немає потреби обмежуватися тільки активним листом. Аналогічно працюють властивості `Columns` і `Rows`.
- `EnableCalculation` - забезпечує можливість включати/відключати автоматичний перерахунок значень комірок робочої книги.
- `EnableSelection` - повертає інформацію/визначає, що може бути виділено у листі: нічого (`xlNoSelection=-4142`), лише незаблоковані комірки (`xlUnlockedCells=1`), усі об’єкти (`xlNoRestrictions=0`). Ця властивість працює тільки для захищених робочих листів.
- `Next` - повертає посилання на наступний лист у робочій книзі, `Previous` - повертає посилання на попередній лист у робочій книзі.
- `PageSetup` - як і в Word, повертає об'єкт `PageSetup`, за допомогою якого можна встановити налаштування параметрів, доступних через меню *Файл -> Параметры*.
- `Protection` - повертає об'єкт `Protection`, за допомогою якого можна заблокувати комірки робочого листа. Налаштування параметрів захисту також виконують і інші властивості, назви яких починаються з `Protection`.

Найважливіші властивості об’єкту Worksheet

- `QueryTables` - містить посилання на колекцію `QueryTables` - набір об'єктів `QueryTable`, які представляють дані, отримані із зовнішніх джерел (як правило, з баз даних).
- `Range` - повертає об'єкт `Range` (діапазон комірок), який в об'єктній моделі Excel займає приблизно таке ж місце, що і однойменний об'єкт в об'єктній моделі Word. Цей об'єкт буде розглядатися нижче.
- `Type` - повертає/надає можливість визначити тип даного листа. Зазвичай, використовуються два типи: `xlWorksheet = -4167` (звичайний лист) і `xlChart = -4109` (діаграма).
- `UsedRange` - повертає об'єкт `Range`, який представляє прямокутну область, що включає всі непусті комірки.
- `Visible` - надає можливість сховати лист (наприклад, якщо він використовується для службових цілей).

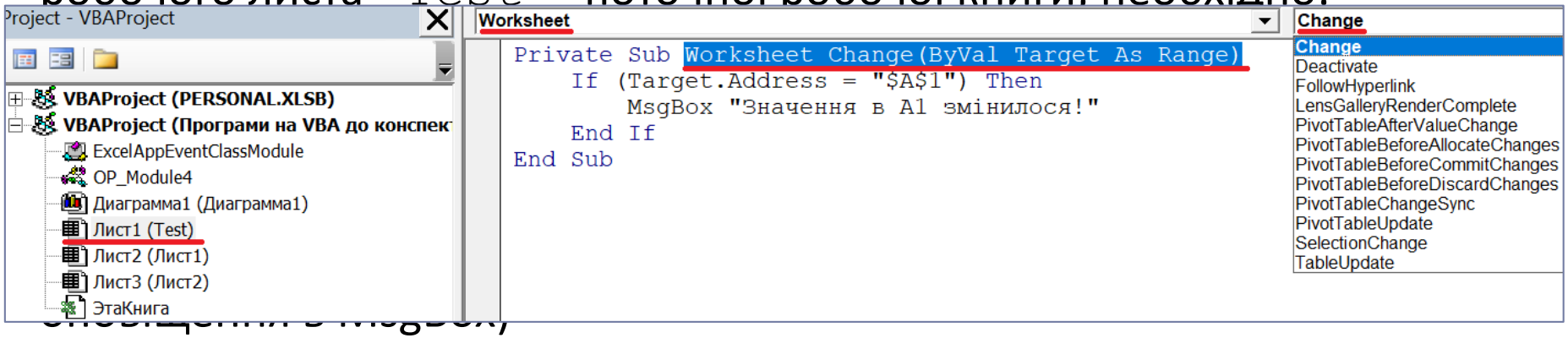
Sub WorksheetProperties()

Найважливіші методи об'єкта Worksheet

- методи `Activate()`, `Calculate()`, `Copy()`, `Paste()`, `Delete()`, `Move()`, `Evaluate()`, `Select()`, `SaveAs()`, `PrintOut()`, `PrintPreview()`, `Protect()`, `Unprotect()` були попередньо розглянуті. Їх відмінність полягає лише в тому, що тепер ці методи можуть застосовуватися для обраного листа.
- `PivotTables()` - повертає колекцію об'єктів `PivotTable` (зведена таблиця).
- `Scenarios()` - повертає колекцію `Scenarios`, що складається з об'єктів `Scenario` (*сценарії*). *Сценарії* - це іменовані набори вхідних даних, які можна використовувати для перевірки різних сценаріїв (різні суми продажів, рівні податків, витрат і т.і.)
- `SetBackgroundPicture()` - надає можливість призначити листу фонове зображення (бажано, щоб воно було напівпрозоре - "водяний знак", інакше на ньому буде важко читати текст комірок).
- `ShowAllData()` - дозволяє показати всі приховані і відфільтровані дані на листі.
- Найважливіша подія об'єкта `Worksheet` - це `Change`. Існує безліч практичних завдань, коли зміна користувачем значення комірки повинна приводити до зміни значення в комірці іншого листа/робочої книги або в базі даних.

Події об'єкта Worksheet

- Найважливіша подія об'єкта `Worksheet` - це `Change`. Існує безліч практичних завдань, коли зміна користувачем значення комірки повинна приводити до зміни значення в комірці іншого листа/робочої книги або в базі даних. Інша ситуація, в якій використовується ця подія - перевірка введеного користувачем значення (наприклад, знову-таки через звернення до бази даних). Процедура обробки події `Change` використовує параметр `Target` - об'єкт `Range`, що представляє комірку, значення якої змінюється. За допомогою властивостей і методів об'єкта `Range` можливо отримати інформацію про змінене значення, стовпець рядок, у якому відбулася зміна і т.і. Наприклад, для організації обробки події зміни значення у комірці з адресою `"A1"` робочого листа `"Test"` поточної робочої книги, необхідно:



Робота з об'єктом Range

- Мабуть, найбільш часто використовуваний об'єкт в ієрархії об'єктної моделі Excel - це об'єкт `Range`. Цей об'єкт може представляти одну комірку, кілька комірок (у тому числі несуміжні комірки або набори несуміжних комірок) або цілий лист. На відміну від Word, у якому для введення даних використовують як об'єкт `Range`, так і об'єкт `Selection`, в Excel все зводиться до об'єкта `Range`:
  - якщо потрібно ввести дані в комірку або відформатувати її, необхідно отримати об'єкт `Range`, який представляє цю комірку;
  - якщо потрібно зробити щось з виділеними комірками, необхідно отримати об'єкт `Range`, який представляє виділення;
  - якщо потрібно щось зробити з діапазоном комірок, знову-таки необхідно отримати об'єкт `Range`, який представляє цей діапазон.
- Найпоширеніший спосіб отримання об’єкту `Range` - скористатися властивістю `Range`, передбаченоюа для об'єктів `Application`, `Worksheet` і самого об'єкта `Range` (якщо необхідно створити новий діапазон на основі вже існуючого). Отримати посилання на об'єкт Range, який представляє комірку A1 робочого листа "Лист1", можна так:

```
Dim oRange As Range
Set oRange = Worksheets("Лист1").Range("A1")
                                        //"A1:A10"
```

Робота з об'єктом Range

- Із застосуванням властивості Range самого об'єкта Range потрібно бути дуже обережним. Справа в тому, що Excel створює на основі об'єкта Range віртуальний лист зі своєю власною нумерацією. Тому такий код:

```
Set oRange1 = Worksheets("Лист1").Range("C1")
Set oRange2 = oRange1.Range("B1")
oRange2.Value = 20
```

вставить значення `20` не в комірку `B1`, як можна було зрозуміти з коду, а в комірку `D1` (тобто `B1` по відношенню до віртуального листу, що починається з `C1`).

- Ще один спосіб створення об’єкту `Range` - скористатися властивістю `Cells`. За допомогою цієї властивості можливо отримати діапазон, що складається тільки з однієї комірки. Наприклад, для отримання посилання на комірку `D1` можна використовувати код:

```
Dim oRange As Range
Set oRange = Worksheets("Лист1").Cells(1, 4)
```

- Щоб отримати діапазон, що складається з декількох комірок, зручно застосовувати властивості `Range` і `Cells` разом:

```
Dim oRange
Set oRange = Range(Cells(1, 1), Cells(5, 3))
```

Робота з об'єктом Range

- Третій спосіб отримання об’єкту Range - скористатися численними властивостями об'єкта Range, які дозволяють змінити поточний діапазон або створити на основі його новий.
- Зазвичай після того, як необхідну комірку знайдено, в неї потрібно щось записати. Для цієї мети використовується властивість Value, наприклад:

```
oRange.Value = "Моє значення"
```

Найбільш вживані властивості об'єкту Range

- `Address` - дозволяє повернути адресу об'єкту `Range`, наприклад, `$A$1:$C$5`. Ця властивість може приймати декілька параметрів - для визначення стилю посилання, абсолютної або відносної адреси для стовпців і рядків і т.і. Властивість доступна тільки для читання.
- `AllowEdit` - дозволяє визначити, чи зможе користувач правити дану комірку (набір комірок) на захищеному аркуші. Використовується для перевірок.
- `Areas` - виключно важлива властивість. Справа в тому, що об'єкт `Range` може складатися з несуміжних діапазонів комірок. Багато методів стосовно до таких діапазонів поводяться абсолютно непередбачувано або просто повертають помилки. Властивість `Areas` дозволяє розбити подібні нестандартні діапазони на набір стандартних. Створені таким чином об'єкти `Range` будуть додані у колекцію `Areas`. Цю властивість можна використовувати і для перевірки "нестандартності" діапазону:

Sub RangeProperties()

- `Borders` - містить посилання на колекцію `Borders`, за допомогою якої можна управляти рамками для діапазону. Приймає параметр `Index`, значення якого вказує яку границю додавати, якщо не вказується будуть додані границі навколо кожної комірки діапазону. Властивість `Borders.LineStyle` визначає тип лінії границі.

Найбільш вживані властивості об'єкту Range

- `Cells` - ця властивість працює аналогічно однойменним властивостям об’єктів `Application` та `Worksheet`, за винятком того, що знову-таки використовується своя власна віртуальна адресація на основі діапазону. Такі ж особливості наявні і у властивостей `Row` і `Rows`, `Column` і `Columns`.
- `Characters` - дозволяє змінити (текст або формат) частини тексту в комірці, не зачіпаючи інші дані.
- `Count` - повертає кількість комірок у діапазоні, може використовуватися для перевірок.
- `CurrentRegion` - дуже зручна властивість, яка може стати в нагоді, наприклад, при копіюванні/експорті даних, отриманих з зовнішнього джерела (коли скільки буде цих даних, нам спочатку невідомо). Вона повертає об'єкт `Range`, який представляє діапазон, оточений порожніми комірками (тобто непорожню область, в яку входить вихідний діапазон/комірка).
- `Dependents` - дозволяє отримати об'єкт `Range` (швидше за все, до складу якого входять несуміжні діапазони), які залежать від комірок вихідного діапазону. Працює тільки для поточного листа - посилання в зовнішніх аркушах цією властивістю не відслідковуються.

Sub RangeProperties()

Найбільш вживані властивості об'єкту Range

- Щоб переглянути зворотну залежність, можна використовувати властивість `Precedents`. Щоб переглянути тільки перший рівень залежностей, можна використовувати властивості `DirectDependents` і `DirectPrecedents`.
- End - ще одна часто використовувана властивість. Вона дозволяє отримати об'єкт `Range`, який представляє останню комірку діапазону. В якій стороні буде вважатися остання комірка, можна визначити за допомогою обов’язкового переданого параметра Direction (`xlDown, xlToLeft, xlToRight, xlUp`).
- `Errors` - властивість, яка через колекцію `Errors` дозволяє отримати доступ до об'єктів `Error`, які представляють виявлені помилки в діапазоні.
- `Font` - як і в Word, ця властивість дозволяє отримати доступ до об'єкта `Font`, за допомогою якого можна налаштувати особливості оформлення тексту в комірці (колір, шрифт, розмір букв і т.і.).
- `FormatConditions` - містить посилання на колекцію об’єктів `FormatCondition`, що дозволяють виконати форматування діапазону при виконанні визначеної умови.

Sub RangeProperties()

Найбільш вживані властивості об'єкту Range

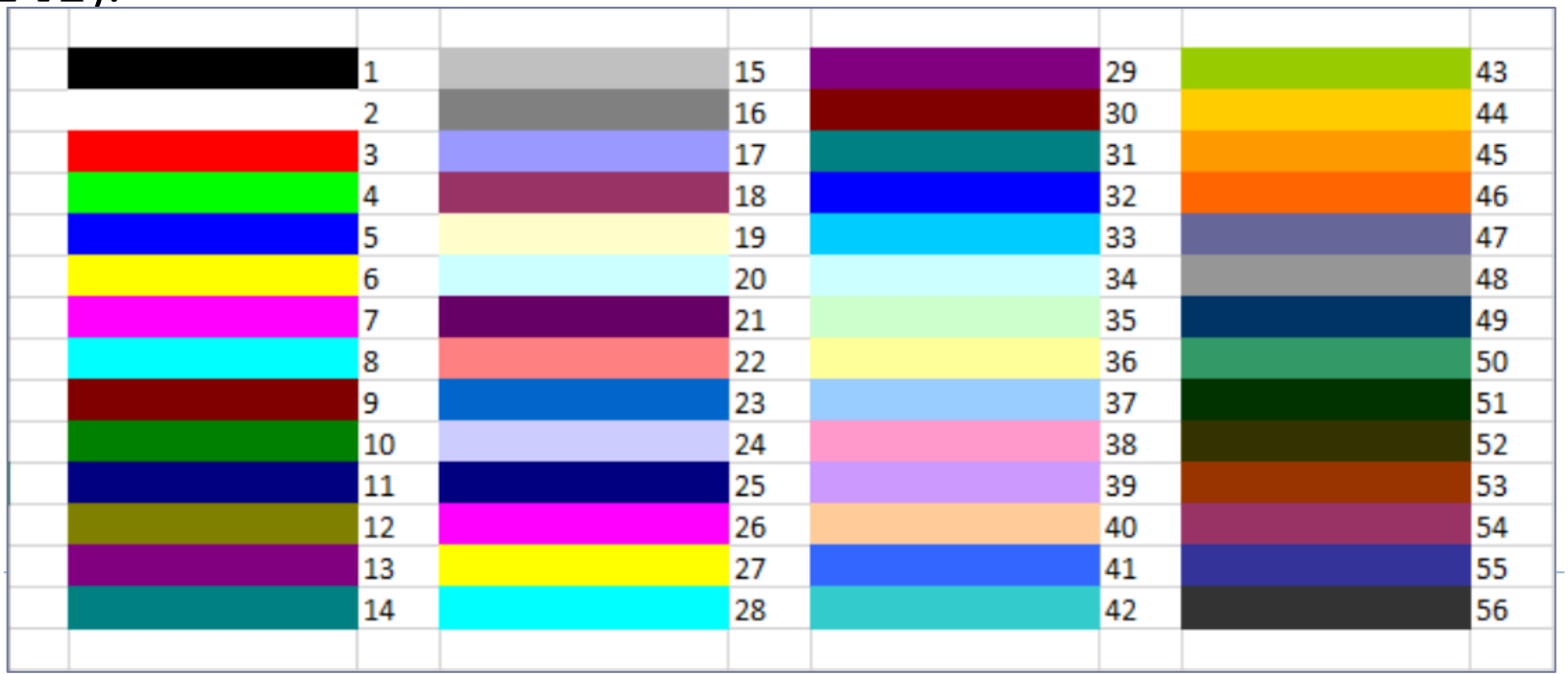
- `Formula` - одна з найважливіших властивостей об'єкта `Range`, доступна і для читання, і для запису. Якщо властивість використовується для читання, то вона повертає текст формули, прописаної у комірці (а не розраховане значення), якщо властивість використовується для запису, то вона дозволяє записати формулу в комірку (наголосимо, що формула повинна вписуватись як текст). Якщо застосувати цю властивість для діапазону, в який входить кілька комірок, то формула буде прописана у всі комірки діапазону. Приклад застосування цієї властивості може виглядати так:
```
Worksheets("Лист1").Range("A3").Formula =
"=$A$1+$A$2"
```
- Якщо формула містить функцію з параметрами, які передаються посиланнями, то значення параметрів повинно конкатенуватись з функцією у вигляді рядка:
```
oRng.Formula = "=FACT(" & oRng.Value & ")"
```
- `FormulaHidden` - надає можливість сховати формули в діапазоні від користувача. Працює тільки на захищених аркушах.
- `HasFormula` - дозволяє перевірити діапазон на наявність обчислюваних значень (формул).

Sub RangeProperties()

Найбільш вживані властивості об'єкту Range

- `Hidden` - дозволяє сховати діапазон. Працюватиме тільки у разі, якщо діапазон включає в себе хоча б один рядок або стовпець цілком, в іншому випадку повернеться помилка.
- `Interior` - повертає об’єкт `Interior` з деякими ознаками форматування діапазону, найчастіше - кольором фону (дозволяє використовувати один з 56 кольорів, значення яких передаються параметру `ColorIndex`, також доступні значення `xlColorIndexAutomatic = -4105 та xlColorIndexNone = -4142`).

Sub RangeProperties()



Найбільш вживані властивості об'єкту Range

- `Item` - дозволяє отримати об'єкт `Range` (комірку або діапазон) вихідного діапазону.
- `Locked` - дозволяє заблокувати комірки діапазону при захисті листа.
- `Name` - надає можливість отримати посилання на спеціальний об'єкт іменованого діапазону `Name`. На графічному екрані з його можливостями можна познайомитися за допомогою меню *Формулы -> Диспетчер имен*. Ця властивість дозволяє звертатися до діапазонів і формул за іменами і дещо нагадує по функціональності об'єкт закладки в Word.
- `Next` - дозволяє перейти на наступну комірку. Якщо лист не захищений, то наступною коміркою буде вважатися комірка праворуч, якщо захищений - то наступна незаблокована комірка.
- `NumberFormat` - надає можливість встановити один з зумовлених форматів для чисел. Відповідає можливостям вкладки *Главная подменю Число* на графічному екрані.
- `Offset` - ця властивість дозволяє отримати новий об'єкт `Range` з певним зміщенням від початкового.
- `Orientation` - дозволяє зорієнтувати текст в комірках. Вказується кут нахилу в градусах.

Sub RangeProperties()

Найбільш вживані властивості об'єкту Range

- `PageBreak` - використовується для програмної вставки розривів сторінки
- Всі властивості, які починаються на `Pivot...`, відносяться до роботи з об'єктом `PivotTable` (зведена таблиця).
- `QueryTable` - дозволяє отримати посилання на об'єкт `QueryTable` - отримані з зовнішнього джерела дані. Ця властивість для об'єкта `Range` дозволяє отримати посилання на об'єкт `QueryTable`, з даними, які знаходиться в поточному діапазоні.
- `Range` - дозволяє створити новий діапазон на основі вже існуючого. Необхідно пам'ятати про особливості нумерації комірок у цьому випадку.
- `Resize` - надає можливість змінити поточний діапазон.
- `ShrinkToFit` - дозволяє автоматично налаштувати розмір тексту в діапазоні таким чином, щоб текст умістився у ширину стовпця.
- `Style` - містить посилання на об'єкт `Style`, який представляє стиль для зазначеного діапазону. На графічному екрані це можна зробити через меню *Главная подменю Стили.*
- `Text` - надає можливість отримати значення першої комірки діапазону у вигляді значення типу `String`. Для об'єкта `Range` ця властивість доступна тільки для читання.

Sub RangeProperties()

Найбільш вживані властивості об'єкту Range

- `Validation` - містить посилання на об'єкт Validation, за допомогою якого можна налаштувати перевірку даних, які вводяться у діапазон.
- `Value` - найбільш часто використовувана властивість об'єкта `Range`. Дозволяє отримати або призначити значення (числове, текстове або будь-яке інше) коміркам діапазону. Точно для тієї ж мети використовується властивість `Value2`, єдина відмінність - ця властивість не підтримує типи даних `Currency` і `Date`.
- `WrapText` - надає можливість включити/відключити перенесення тексту на наступний рядок в комірках діапазону

Sub RangeProperties()

Найважливіші методи об’єкта Range

- `Activate()` - виділяє поточний діапазон і встановлює курсор введення на його першу комірку.
- `AddComment()` - надає можливість додати коментар до комірки. Цей метод можна викликати тільки для діапазону, що складається з однієї комірки. Те ж саме на графічному екрані *можна зробити за допомогою меню* Рецензирование -> Создать примечание.
- `AutoFill()` - надає можливість використовувати автозаповнення для діапазону (наприклад, якщо перші два осередки будуть заповнені як `1` і 2, то далі в автоматичному режимі буде продовжено: `3, 4, 5` і т.і.)

Sub RangeMethods()

- `AutoFit()` - дозволяє автоматично змінити ширину всіх стовпців і висоту всіх рядків у діапазоні, щоб в них вмістився текст комірок. Можна застосовувати тільки до діапазонів, які складаються з набору стовпців (повністю) або з набору осередків (також повністю).
- `BorderAround()` - надає можливість помістити діапазон в рамку з вибраними (опціональними) параметрами: `LineStyle` - стиль границі - приймає значення, як параметр `LineStyle` властивості `Borders` об’єкту `Range`; `Weight` - товщина границі - приймає значення енумератора `XlBorderWeight (xlHairline, xlThin, xlMedium, xlThick)`, `ColorIndex` - визначається цілим числом

Найважливіші методи об’єкта Range

- методи `Clear`... дозволяють очистити вміст діапазону: `Clear` - від всього (значень, форматування, коментарів і т.і.), `ClearContents` - тільки від значень, `ClearFormats` - тільки від форматування, `ClearComments` - тільки від коментарів.
- `Consolidate()` - надає можливість злити дані декількох діапазонів (у тому числі на різних аркушах) в один діапазон, використовуючи при цьому обрану вами агрегатну функцію. Параметри метода (всі опціональні): `Sources` - масив адрес діапазонів у `R1C1`-стилі, записаних як рядки, `Function` - агрегатна функція, визначення константою-елементом енумератора `XlConsolidationFunction` (найбільш популярні константи : `xlSum, xlAverage, xlCount, xlMax, xlMin, xlAverage), TopRow – True` - для консолідації даних на основі заголовків стовпців у верхньому рядку діапазонів консолідації, `False` (за замовчуванням) - консолідувати дані за позицією, `LeftColumn` - `True` - для консолідації даних на основі заголовків рядків у лівій колонці діапазонів консолідації, `False` (за замовчуванням) - консолідувати дані за позицією, `CreateLinks` - `True` - консолідація використовує посилання на робочий лист, `False` (за замовчуванням) - консолідація копіює дані.

Sub RangeMethods()

Найважливіші методи об’єкта Range

- `Copy()` - копіює діапазон в інше місце, єдиним параметром методу є `Destination` - діапазон (об’єкт `Range`), у який буде виконано копіювання вихідного діапазону. Якщо параметр `Destination` не вказаний, вихідний діапазон копіюється в буфер обміну.

```
Set oRng = oWbk.Worksheets("Лист1").Range("F1:F5")
oWbk.Worksheets("Лист1").Range("E1:E5").Copy
Destination:=oRng
```

- `Cut()` - вирізає діапазон у буфер обміну та вставляє його у діапазон, вказаний параметром `Destination`. Якщо параметр `Destination` не вказаний, вихідний діапазон вирізається в буфер обміну.

```
Set oRng = oWbk.Worksheets("Лист1").Range("E1:E5")
oRng.Cut
Destination:=oWbk.Worksheets("Лист2").Range("A1:A5")
```

Sub RangeMethods()

- Необхідно зазначити, що методу `Paste()` для об’єкту `Range` не існує, тобто операції копіювання та вставки або вирізання та вставки об’єднуються у відповідних методах. Існує метод `PasteSpecial()` (буде розглянутий далі), який дозволяє вставляти з буферу обміну скопійований (вирізаний) у нього діапазон, але він призначений для вставки форматів, коментарів, формул, значень і т.і. Тобто при звичайному копіюванні (вирізанні) та вставці він не задіюється.

Найважливіші методи об’єкта Range

- `CopyFromRecordset()` - дозволяє вставити дані з об'єкта ADO `Recordset` на лист Excel, починаючи з верхнього лівого кута зазначеного діапазону.
- `DataSeries()` - дозволяє збільшити значення дати в діапазоні на вказаний інтервал часу. Наприклад, якщо в діапазоні вказане перше січня, то за допомогою цього методу можна згенерувати перше число будь-якого іншого місяця.
- `Delete()` - видаляє дані поточного діапазону, необов'язковий параметр `Shift` приймає значення, які вказують куди будуть зміщуватися комірки, які замінять видалені комірки (вгору - `xlShiftUp`, або вліво - `xlShiftToLeft`).
- `Dirty()` - позначає комірки діапазону як "брудні". Такі комірки будуть перераховані при наступному ж перерахунку. Зазвичай використовується, коли Excel сам не може здогадатися, що їх потрібно перерахувати. Перерахувати комірки діапазону можна і примусово - за допомогою методу `Calculate()`.
- методи `Fill... (FillDown(), FillUp(), FillLeft(), FillRight())` дозволяють розмножити одне і те ж значення (розміщене у першій комірці діапазону з урахуванням напрямку) по коміркам діапазону в зазначеному напрямку.

Sub RangeMethods()

Найважливіші методи об’єкта Range

- `Find()` - дозволяє виконувати пошук в комірках діапазону і повернути новий об'єкт Range, який представляє першу комірку, у якій було знайдено значення, що шукалося. Обов’язковий параметр `What` вказує предмет пошуку у вигляді рядка або значення будь-якого типу даних Excel. У цього методу є необов'язкові параметри: `After` - дозволяє почати пошук в діапазоні, починаючи з комірки діапазону, вказаної як значення цього параметру у вигляді об’єкту `Range`; `LookIn` - визначає тип даних, що шукаються (значення - `xlValues` (за замовчуванням), формули - `xlFormulas` або коментарі `xlComments`); `LookAt` - дозволяє визначити шаблон пошуку повинен співпадати з повним (`xlWhole`) або частковим (`xlPart`) (за замовчуванням) вмістом комірки; `SearchOrder` - визначає спосіб пошуку - по рядкам (`xlByRows`) (за замовчуванням) або по стовпцям (`xlByColumns`); `SearchDirection` - визначає напрямок пошуку - вперед (`xlNext`) (за замовчуванням) або назад (`xlPrevious`); `MatchCase` - визначає чутливість до регістру (`True/False`) (за замовчуванням `False`); `SearchFormat` - дозволяє виконувати пошук шаблону з зазначеними ознаками форматування (`True/False`), спочатку шаблон пошуку повинен бути встановленим у властивість `Application.FindFormat`.

Sub RangeMethods()

Найважливіші методи об’єкта Range

- Методи `FindNext()` і `FindPrevious()` дозволяють продовжити пошук, розпочатий методом `Find()`, в різних напрямках, приймають параметр `After`, у який необхідно записати посилання на об’єкт `Range`, знайдений попереднім пошуком.
- `Insert()` - виконує вставку комірки (або діапазон комірок) в вихідний діапазон, необов'язковий параметр `Shift` приймає значення, які вказують куди будуть зміщуватися комірки, на місце яких виконується вставка (вниз - `xlShiftDown`, вправо - `xlShiftToRight`).
- `Justify()` - дозволяє рівномірно розподілити текст у діапазоні. Якщо у даний діапазон він не вміщується, він буде поширений на сусідні нижчі комірки (з перезаписом їх значень). Перед перезаписом з’явиться діалогове вікно його підтвердження. Для запобігання його появи перед викликом методу Justify потрібно відключити появу повідомлень (`Application.DisplayAlerts`), після виклику методу .
- `Merge()` - дозволяє об’єднати всі комірки діапазону в одну. При цьому залишиться значення тільки однієї - верхньої лівої комірки, перед завершенням виконання методу з’явиться діалогове вікно з повідомленням (таке попередження може бути відключене як у методі Justify()). Розбити назад таку об’єднану комірку на кілька звичайних можна за допомогою методу `UnMerge()`.

Sub RangeMethods()

Найважливіші методи об’єкта Range

- `PasteSpecial()` – дозволяє вставити вміст буферу обміну із зазначенням спеціальних параметрів вставки: Paste - визначає, що буде вставлятися (найбільш поширені варіанти - xlPasteValues - тільки значення, xlPasteFormats - тільки ознаки форматування, xlPasteFormulas - тільки формули, xlPasteAll - все); Operation - визначає тип операції над даними, які вставляються та даними у діапазоні у який виконується вставка (з додаванням - xlPasteSpecialOperationAdd, з множенням - xlPasteSpecialOperationMultiply, відніманням - xlPasteSpecialOperationSubtract, діленням - xlPasteSpecialOperationDivide, вставка без операцій - xlPasteSpecialOperationNone). Параметр SkipBlanks дозволяє пропускати пусті комірки у вихідному діапазоні і не копіювати їх у цільовий діапазон (значення True). Параметр Transpose зі значенням True дозволить дані діапазону, розміщеного по рядкам розмістити по стовпцям і навпаки.
- `PrintOut()` і `PrintPreview()` - дозволяють вивести діапазон на друк або відкрити режим перегляду перед друком.

Sub RangeMethods()

Найважливіші методи об’єкта Range

- `Replace()` - дозволяє виконувати пошук і заміну значень в комірках діапазону і повернути значення Boolean, яке вказує, чи відбулась хоча б одна заміна (True - якщо так). Обов’язковий параметр What вказує шаблон пошуку у вигляді рядка, обов’язковий параметр Replacement містить рядок на який буде замінений знайдений. Необов'язкові параметри: LookAt, SearchOrder, SearchFormat - працюють так, як у методі Find(). Необов’язковий параметр ReplaceFormat - дозволяє змінювати ознаки форматування, спочатку шаблон заміни повинен бути встановленим у властивість Application.ReplaceFormat.
- `Select()` - надає можливість виділити зазначений діапазон. Об'єкта Selection в Excel немає - замість нього є можливість отримати об'єкт Range, який представляє виділену область.
- `Show()` - дозволяє прокрутити екран таким чином, щоб показати вказаний діапазон.
- `ShowDependents()` - вказує стрілками осередки осередків, які залежать від зазначеного діапазону (тільки перший рівень залежності) або прибирає ці стрілки. Зворотний метод - `ShowPrecedents()`.
- `ShowErrors()` - показує джерело помилки для зазначеної комірки.

Sub RangeMethods()

Найважливіші методи об’єкта Range

- Sort() - надає можливість сортування значень комірок діапазону. Існує велика кількість необов'язкових параметрів для налаштування сортування, опишемо найбільш використовувані. Параметр Key1 містить перше поле для сортування - об’єкт Range, значення у якому повинні бути відсортованими, параметр Order1 вказує порядок сортування (по зростанню - xlAscending, або по зменшенню - xlDescending). Параметри Key2, Order2 та Key3, Order3 працюють аналогічно Key1, Order1 і визначають поля та порядок сортування, по яким буд виконуватись сортування, якщо в попередніх полях значення однакові. Параметр Header визначає, чи вважати перший рядок рядком заголовків (наявні значення xlYes, xlNo, xlGuess - останнє для інтерактивного визначення). Параметр MatchCase визначає, чи враховувати регістр при сортуванні (його значення True/False).

Sub RangeMethods()

Найважливіші методи об’єкта Range

- SpecialCells() - дозволяє повернути об'єкт Range, що включає в себе всі комірки певного типу. Приймає обов’язковий параметр Type, який визначає тип комірок (порожні - xlCellTypeBlanks, з коментарями - xlCellTypeComments, останні у діапазоні - xlCellTypeLastCell, з константами - xlCellTypeConstants, з формулами - xlCellTypeFormulas і т.і.), а також необов’язковий праметр Value, який визначає тип значення комірок з типом xlCellTypeConstants або xlCellTypeFormulas (доступні: значення з помилками - xlErrors, логічні (булеві) значення - xlLogical, числа xlNumbers, текст - xlTextValues). Наприклад, щоб повернути об'єкт Range, що складається з усіх порожніх комірок діапазону, можна використовувати код:

```
Set oRange2 = oRange.SpecialCells(xlCellTypeBlanks)
oRange2.Select 'візуалізуємо виділення
```

Sub RangeMethods()

Використання інших об’єктів об’єктної моделі Excel

- Об'єкт `CellFormat` дозволяє налаштовувати форматування комірок Excel - шрифт, рамки, кольорове оформлення і т.і. При цьому можна використовувати цей об'єкт для копіювання оформлення з інших осередків, для пошуку і заміни з оформлення і т.і. Умовне форматування можна налаштувати за допомогою об'єкта `FormatCondition`.
- `ListObject` - об'єкт, який призначений для роботи зі списками - наборами взаємопов'язаних даних (наприклад, списками співробітників).
- Об'єкт `Validation` дозволяє налаштувати перевірку даних, які вводяться користувачем.

Sub RangeMethods()