

Techniki programowania



Automatyzacja operacji, dodatki

Tworzenie procedur zdarzeniowych
Zdarzenia obiektów, systemowe i aplikacji
Tworzenie dodatków

Zdarzenia i procedury zdarzeniowe

Zdarzenie – sygnał informujący o wystąpieniu określonej sytuacji rozpoznawanej i rejestrowanej przez system komputerowy. Może być generowane przez system lub działania podejmowane przez użytkownika oprogramowania. Typ zdarzenia określa jego wyzwalacz (akcja lub sytuacja, która je wywołała).

VBA wyróżnia dwie kategorie zdarzeń: związane z konkretnym obiektem (np. Workbook, Worksheet) oraz zdarzenia systemowe.

Przykłady zdarzeń w VBA

- otwieranie lub zamykanie skoroszytu,
- aktywacja lub dezaktywacja arkusza,
- zmiana zawartości komórki arkusza,
- kliknięcie klawiszem myszy,
- drukowanie skoroszytu,
- wystąpienie określonej chwili czasowej,
- naciśnięcie klawisza na klawiaturze.

Procedura zdarzeniowa (procedura obsługi zdarzenia) – procedura wykonywana automatycznie gdy system zarejestruje wystąpienie określonego zdarzenia.

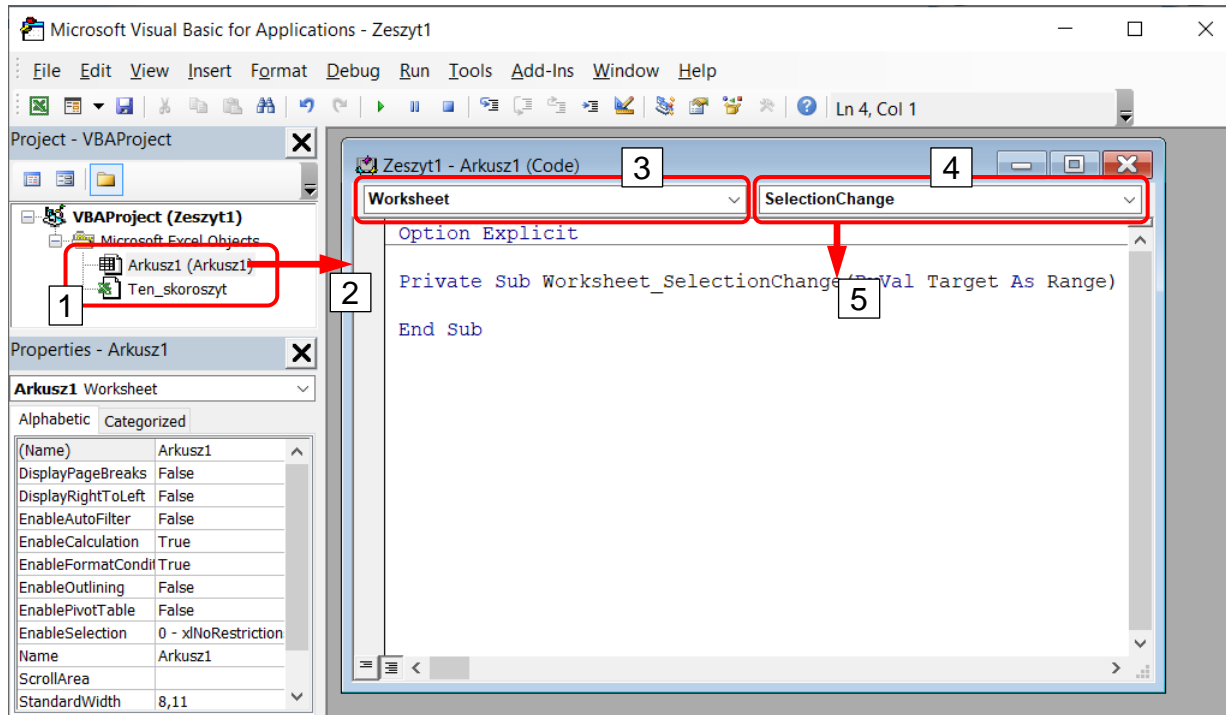
Wybrane zdarzenia obiektu Workbook

Activate	aktywacja skoroszytu
BeforeClose	zamykanie skoroszytu
BeforePrint	drukowanie skoroszytu
BeforeSave	zapisywanie skoroszytu
Deactivate	dezaktywacja skoroszytu
NewSheet	dodawanie nowego arkusza do skoroszytu
Open	otwieranie skoroszytu
SheetActivate	aktywacja arkusza w skoroszytcie
SheetBeforeDoubleClick	podwójne kliknięcie komórki w skoroszytcie
SheetBeforeRightClick	kliknięcie komórki w skoroszytcie prawym przyciskiem myszy
SheetChange	zmiana komórki w skoroszytcie
SheetDeactivate	dezaktywacja arkusza w skoroszytcie
SheetSelectionChange	zmiana zaznaczenia
WindowActivate	aktywacja okna skoroszytu
WindowDeactivate	dezaktywacja okna skoroszytu

Wybrane zdarzenia obiektu Worksheet

Activate	aktywacja arkusza
BeforeDelete	usunięcie arkusza
BeforeDoubleClick	podwójne kliknięcie komórki w arkuszu
BeforeRightClick	kliknięcie komórki w arkuszu prawym przyciskiem myszy
Calculate	przeliczenie arkusza
Change	zmiana komórki w arkuszu
Deactivate	dezaktywacja arkusza
FollowHyperlink	wybór (kliknięcie) odnośnika
SelectionChange	zmiana zaznaczenia

Tworzenie procedury zdarzeniowej



1. Wybór obiektu, którego zdarzenie będzie definiowane.
2. Podwójne kliknięcie otwiera okno kodu obiektu (w przykładzie obiekt „Arkusz1”).
3. Wybór obiektu z listy rozwijalnej („General” zawiera stałe/zmienne modułowe).
4. Wybór zdarzenia obiektu z listy rozwijalnej.
5. Szkielet procedury zdarzenia (nagłówek, argumenty, **End Sub**).

Zdarzenia obiektu Workbook – automatyczny backup

Tworzenie kopii pliku po otwarciu dokumentu (plik zapisywany w bieżącym folderze pod nazwą „Backup_<data-godzina>_nazwa_pliku”).

```
Private Sub Workbook_Open()  
    Dim d As Date  
    Dim n As String  
    d = Now  
    n = "\Backup_" & Year(d) & Month(d) & Day(d) & _  
        Hour(d) & Minute(d) & Second(d) & "_" & Ten_skoroszyt.Name  
    Ten_skoroszyt.SaveCopyAs Ten_skoroszyt.Path & n  
End Sub
```

Ten_skoroszyt reprezentuje aktualny dokument

Now zwraca aktualną datę i godzinę

Year, Month, Day zwracają rok, miesiąc i dzień z podanego argumentu

Hour, Minute, Second zwracają godzinę, minutę i sekundę z podanego argumentu

SaveCopyAs (metoda obiektu Workbook) zapisuje kopię dokumentu

Zdarzenia obiektu Workbook – blokada zapisu

Zablokowanie zapisu dokumentu pod nową nazwą (opcja „Zapisz jako...”)

```
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, _  
                                Cancel As Boolean)  
  
    If SaveAsUI Then  
        MsgBox "Zmiana nazwy niedozwolona", vbExclamation  
        Cancel = True  
    End If  
End Sub
```

Argumenty procedury zdarzeniowej BeforeSave

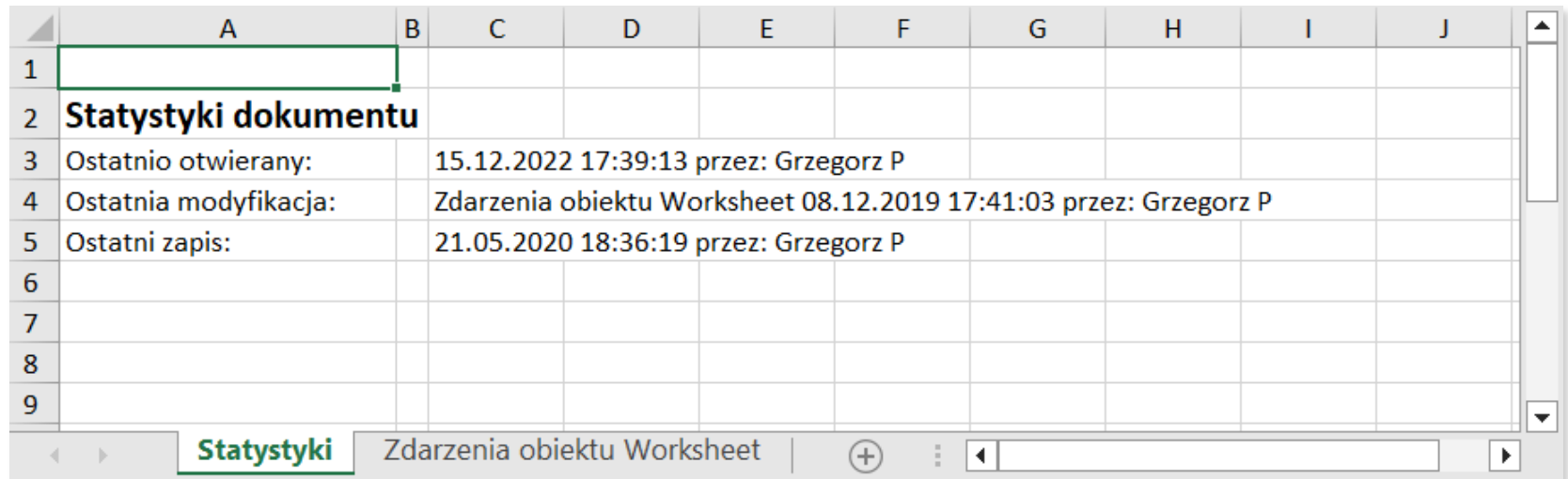
- SaveAsUI otrzymuje wartość `True` jeżeli będzie wyświetlone okno wyboru pliku (użytkownik wybrał opcję „Zapisz jako”),
- Cancel określa czy operacja zapisu ma być przerwana: wartość `False` (domyślnie) zapis jest kontynuowany, `True` zapis zostaje przerwany.

Zdarzenia obiektu Workbook – statystyki dokumentu (1)

Automatyzacja procesu realizacji statystyk dokumentu:

- Ostatnie otwarcie (data, godzina i użytkownik)
- Ostatnia modyfikacja (nazwa arkusza, data, godzina i użytkownik)
- Ostatni zapis (data, godzina i użytkownik)

Wykorzystane zdarzenia: `Open`, `SheetChange`, `BeforeSave`



The screenshot shows an Excel spreadsheet with the following data:

	A	B	C	D	E	F	G	H	I	J
1										
2	Statystyki dokumentu									
3	Ostatnio otwierany:		15.12.2022 17:39:13 przez: Grzegorz P							
4	Ostatnia modyfikacja:		Zdarzenia obiektu Worksheet 08.12.2019 17:41:03 przez: Grzegorz P							
5	Ostatni zapis:		21.05.2020 18:36:19 przez: Grzegorz P							
6										
7										
8										
9										

The spreadsheet interface includes a tab labeled "Statystyki" and a sheet name "Zdarzenia obiektu Worksheet".

Zdarzenia obiektu Workbook – statystyki dokumentu (2)

Zapis informacji o ostatnim otwarciu i zapisie dokumentu.

```
Private Sub Workbook_Open()  
    Worksheets("Statystyki").Range("C3").Value = Now & _  
    " przez: " & Application.UserName  
End Sub  
  
Private Sub Workbook_BeforeSave(ByVal SaveAsUI As Boolean, _  
                                Cancel As Boolean)  
    Worksheets("Statystyki").Range("C5").Value = Now & _  
    " przez: " & Application.UserName  
End Sub
```

Uwaga: program należy uzupełnić o sprawdzenie istnienia arkusza „Statystyki” (może być tworzony w chwili otwarcia).

Zdarzenia obiektu **Workbook** – statystyki dokumentu (3)

Zapis informacji o ostatniej modyfikacji dokumentu.

```
Private Sub Workbook_SheetChange(ByVal Sh As Object, _  
                                ByVal Target As Range)  
  
    If Sh.Name <> "Statystyki" Then  
        Worksheets("Statystyki").Range("C4").Value = Sh.Name & _  
        " " & Now & " przez: " & Application.UserName  
  
    End If  
  
End Sub
```

Argumenty procedury zdarzeniowej SheetChange

- Sh reprezentuje arkusz, który został zmodyfikowany
- Target określa zakres arkusza, który uległ modyfikacji.

Zdarzenia obiektu Workbook – tworzenie arkusza (1)

Po dodaniu arkusza (opcja „Nowy arkusz”, zdarzenie: `NewSheet`) zostanie on wstępnie wypełniony zdefiniowaną zawartością.

Wykorzystane metody i funkcje:

- `Merge` metoda obiektu `Range`, scala komórki w zakresie,
- `Borders` metoda obiektu `Range`, ustala obramowania komórek,
- `AutoFit` metoda obiektu `Range`, dopasowuje szerokość kolumn i wysokość wierszy do zawartości komórek w zakresie,
- `WeekdayName` funkcja zwraca nazwę dnia tygodnia.

The image shows two overlapping Excel worksheets. The foreground sheet, 'Statystyki', contains the following data:

A	B	C	D	E	F
1					
2	Statystyki dokumentu				
3	Ostatnio otwierany:	16.12.2022 19:04:28 przez: Grzegorz P			
4	Ostatnia modyfikacja:	Arkusz1 16.12.2022 19:04:38 przez: Grzegorz			
5	Ostatni zapis:	15.12.2022 18:46:22 przez: Grzegorz P			
6					
7					
8					
9					
10					

The background sheet, 'Arkusz2', contains the following data:

1	A	B	C	D	E
1		Sprzedaż			
2	Dzień	sztuk	wartość netto	wartość brutto	
3	poniedziałek				
4	wtorek				
5	środa				
6	czwartek				
7	piątek				
8	sobota				
9	niedziela				
10					

The sheet tab bar at the bottom shows 'Statystyki' and 'Arkusz2'. A red box highlights the '+' button in the sheet tab bar, with a red arrow pointing to the 'Arkusz2' tab.

Zdarzenia obiektu Workbook – tworzenie arkusza (2)

```
Private Sub Workbook_NewSheet(ByVal Sh As Object)
    Dim k As Range
    Dim i As Integer
    With Sh
        .Range("A1").Value = "Dzień"
        .Range("A1:A2").Merge
        ...
        i = 1
        For Each k In .Range("A3:A9")
            k.Value = WeekdayName(i, , vbMonday)
            i = i + 1
        Next
        .Range("A1:D9").Borders(xlEdgeTop).LineStyle = xlDouble
        ...
        .Columns("A:D").AutoFit
        .Range("B3").Activate
    End With
End Sub
```

Zdarzenia obiektu Worksheet – podwójne kliknięcie

Procedura włącza/wyłącza pogrubienie tekstu w komórce po podwójnym kliknięciu lewym przyciskiem myszy (zdarzenie `BeforeDoubleClick`)

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range,  
                                         Cancel As Boolean)  
  
    Target.Font.Bold = Not Target.Font.Bold  
  
    Cancel = True  
  
End Sub
```

Argumenty procedury zdarzeniowej `BeforeDoubleClick`

- `Target` reprezentuje komórkę arkusza, która została kliknięta,
- `Cancel` określa czy domyślna operacja związana z podwójnym kliknięciem (przejdzie w tryb edycji komórki) ma być przerwana: wartość `False` (domyślnie) operacja jest kontynuowana, `True` przejście w tryb edycji zostaje anulowany.

Uwaga: operacja zmiany stylu czcionki jest tylko przykładem. Przedstawiony schemat można wykorzystać do automatyzacji dowolnej, często wykonywanej operacji.

Zdarzenia obiektu Worksheet – kontrola danych

Procedura uniemożliwia wprowadzenie wartości, które nie są liczbami w zakresie arkusza „C3:I12” (zdarzenie Change)

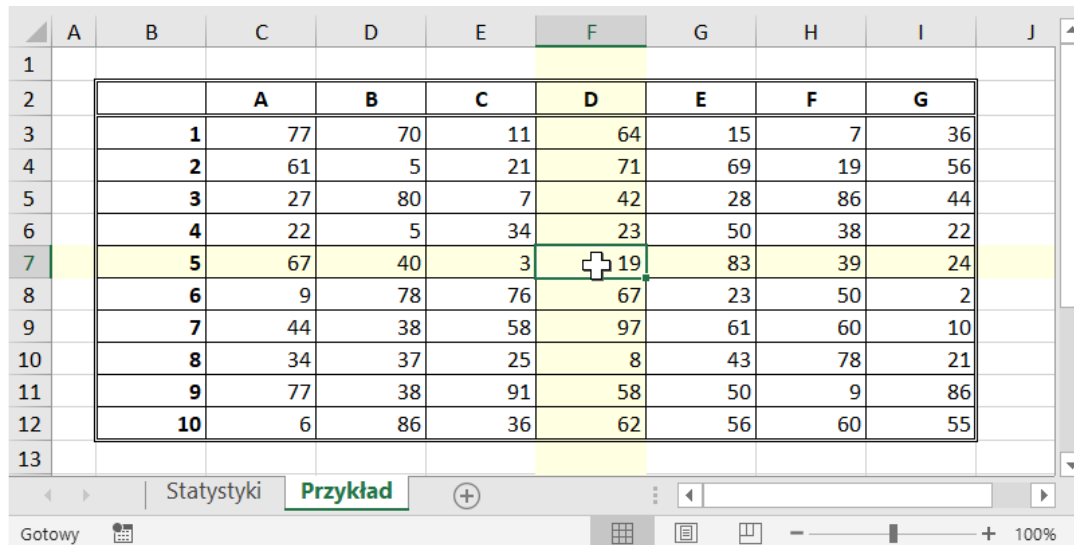
```
Private Sub Worksheet_Change(ByVal Target As Range)  
    If Not Intersect(Range("C3:I12"), Target) Is Nothing Then  
        If Not IsNumeric(Target) Then  
            MsgBox "Komórki tabeli muszą zawierać liczby", vbCritical  
            Target.ClearContents  
            Target.Activate  
        End If  
    End If  
End Sub
```

Uwaga: instrukcja `Intersect(Range("C3:I12"), Target)` określa część wspólną zakresu „C3:I12” oraz parametru `Target` (modyfikowana komórka). Jeżeli wynik nie jest pusty (`Not ... Is Nothing`) `Target` znajduje się w zdefiniowanym zakresie.

Zdarzenia obiektu Worksheet – wskazanie komórki

Procedura wskazuje aktualnie wybraną komórkę arkusza przez zmianę tła wiersza i kolumny odpowiadających tej komórce (zdarzenie `SelectionChange`).

```
Private Sub Worksheet_SelectionChange(ByVal Target As Range)  
    Cells.Interior.ColorIndex = xlNone  
    Columns(ActiveCell.Column).Interior.Color = RGB(255, 255, 225)  
    Rows(ActiveCell.Row).Interior.Color = RGB(255, 255, 225)  
End Sub
```



	A	B	C	D	E	F	G	H	I	J
1										
2		A	B	C	D	E	F	G		
3	1	77	70	11	64	15	7	36		
4	2	61	5	21	71	69	19	56		
5	3	27	80	7	42	28	86	44		
6	4	22	5	34	23	50	38	22		
7	5	67	40	3	19	83	39	24		
8	6	9	78	76	67	23	50	2		
9	7	44	38	58	97	61	60	10		
10	8	34	37	25	8	43	78	21		
11	9	77	38	91	58	50	9	86		
12	10	6	86	36	62	56	60	55		
13										

Zdarzenia systemowe – zdarzenie niezwiązane z konkretnym obiektem, mają charakter globalny są uruchamiane niezależnie od otwartego dokumentu i aktywowanego arkusza.

Zdarzenia systemowe w VBA

- **OnKey** – naciśnięcie klawisza na klawiaturze,
- **OnTime** – wystąpienie określonej chwili czasowej,
- **OnUndo** – wybór opcji „Cofnij” (Undo),
- **OnRepeat** – wybór opcji „Powtórz” (Repeat).

Zdarzenia systemowe nie są związane z konkretnym obiektem. Procedury obsługi takich zdarzeń należy definiować w zwykłych modułach podobnie jak makra (procedury) i funkcje arkuszowe. W celu ustawienia zdarzenia systemowego należy użyć odpowiedniej metody obiektu Application.

Uwaga: Zdarzenie systemowe jest aktywne od chwili ustawienia do zakończenia pracy aplikacji, nawet jeżeli Workbook, w którym znajduje się kod obsługi zdarzenia jest zamknięty. Jeżeli zdarzenie ma być aktywne tylko w jednym Workbooku należy je ustawiać i wyłączać w momencie otwierania/zamykania (zdarzenia Open/Close) lub aktywowania i deaktywowania (zdarzenia Activate/Deactivate) dokumentu.

Zdarzenia systemowe OnTime i OnKey

`Application.OnTime EarliestTime, Procedure, LatestTime, Schedule`

`EarliestTime` - `DateTime`, czas wystąpienia zdarzenia

`Procedure` - `String`, nazwa procedury uruchamianej gdy wystąpi zdarzenie

`LatestTime` - `DateTime`, argument opcjonalny, maksymalny czas wystąpienia zdarzenia, po tej chwili zdarzenie będzie anulowane

`Schedule` - `Boolean`, argument opcjonalny, `True` ustawia zdarzenie, `False` anuluje zdarzenia (domyślnie `True`)

`Application.OnKey Key, Procedure`

`Key` - `String`, klawisz, który aktywuje zdarzenie

`Procedure` - `String`, nazwa procedury uruchamianej gdy wystąpi zdarzenie

Wybrane kody klawiszy:

<code>{BS}</code> - Backspace	<code>{DEL}</code> - Delete	<code>{END}</code> - End
<code>{ESC}</code> - Escape	<code>{HOME}</code> - Home	<code>{PGDN}</code> , <code>{PGUP}</code> - Page Down/Up
<code>~</code> - Enter	<code>{TAB}</code> - Tabulacja	<code>{F1}</code> ... <code>{F15}</code> - funkcyjne
<code>{DOWN}</code> , <code>{UP}</code> , <code>{LEFT}</code> , <code>{RIGHT}</code> - strzałki	<code>+</code> , <code>^</code> , <code>%</code>	- Shift, Ctrl, Alt

Przypomnienie o wysłaniu raportu wyświetlane o określonej godzinie

```
Public Sub Przypomnienie()  
    Dim odp As Integer  
    odp = MsgBox("Wysłałeś raport?", vbQuestion + vbYesNo)  
    If odp <> vbYes Then  
        Application.OnTime Now + TimeValue("00:05"), "Przypomnienie"  
    End If  
End Sub
```

Aktywacja (przy otwieraniu dokumentu)

```
Private Sub Workbook_Open()  
    Application.OnTime TimeValue("14:45"), "Przypomnienie"  
End Sub
```

Dezaktywacja (przy zamykaniu dokumentu)

```
Private Sub Workbook_BeforeClose(Cancel As Boolean)  
    Application.OnTime TimeValue("14:45"), "Przypomnienie", , False  
End Sub
```

Zdarzenia aplikacji – zdarzenia związane z obiektem Application, mają charakter globalny, są uruchamiane niezależnie od otwartego dokumentu i aktywowanego arkusza.

Wybrane zdarzenia obiektu Application

- Okno aplikacji: WindowActivate, WindowDeactivate, WindowResize,
- Otwarte dokumenty: WorkbookActivate, WorkbookAfterSave, WorkbookBeforeClose, WorkbookBeforeSave, WorkbookDeactivate, WorkbookNewSheet, WorkbookOpen.
- Arkusze dokumentów: SheetActivate, SheetBeforeDelete, SheetCalculate, SheetChange, SheetDeactivate.
- Operacje wykonane w programie: AfterCalculate, NewWorkbook.

Zdarzenia aplikacji przypominają zdarzenia obiektów Workbook i Worksheet, mają jednak charakter globalny i podobnie jak zdarzenia systemowe są wykonywane dla każdego dokumentu i arkusza (od chwili ustawienia do zakończenia pracy aplikacji).

Uwaga: Jeżeli zdarzenie aplikacji ma być uruchamiane domyślnie na danym stanowisku jego kod można umieścić w arkuszu PERSONAL lub w dodatku (patrz s.23). W obydwu przypadkach zdarzenie będzie tworzone przy pierwszym uruchomieniu aplikacji i pozostanie aktywne do jej zakończenia.

Tworzenie zdarzenia aplikacji

Tworzenie zdarzenia aplikacji

1. W module istniejącego obiektu (arkusz, skoroszyt, własna klasa) zdefiniować zmienną obiektową klasy **Application** z dyrektywą **WithEvents**.
2. Przypisać standardowy obiekt **Application** do utworzonej zmiennej (zmienna i Application będą tym samym obiektem, każda operacja wykonana na zmiennej będzie oddziaływać na standardowy obiekt Application).
3. Z listy **Object** wybrać obiekt reprezentowany przez zmienną (dostępny po zdefiniowaniu obiektu z klauzulą **WithEvents**).
4. Z listy **Procedure** wybrać zdarzenie i zdefiniować jego kod.

Zdefiniowane operacje będą wykonywane po wstąpieniu wybrane zdarzenie od chwili przypisania wartości zmiennej (p.2) do zamknięcia skoroszytu zawierającego kod zdarzenia.

```
Microsoft Visual Basic for Applications - wyklad05.xlsm - [Ten_skoroszyt (Code)]
File Edit View Insert Format Debug Run Tools Add-Ins Window Help
Project - VBAProject
VBAProject (wyklad05.xlsm)
  Microsoft Excel Objects
    Arkusz1 (Statystyki)
    Arkusz14 (Zdarzenia)
    Ten_skoroszyt
  Modules
    Module1

Public WithEvents App As Application

' Zdarzenie uruchamiane przy otwieraniu nowego Workbook-a
Private Sub App_WorkbookOpen(ByVal Wb As Workbook)
    Dim sh As Worksheet
    For Each sh In Wb.Worksheets
        If (sh.Name = "dane") Then
            Dim odp As Integer
            Dim z As Range
            odp = MsgBox("Otwierany dokument zawiera arkusz DANE", vbYesNo)
            On Error Resume Next
            If (odp = vbYes) Then
                sh.Activate
                Set z = sh.Cells.SpecialCells(xlCellTypeConstants)
                z.Font.Color = vbRed
            End If
        End If
    Next
End Sub

Private Sub Workbook_Open()
    ' ustawienie wartości zmiennej App na Application
    ' po wykonaniu tej operacji zdarzenia obiektu App
    ' będą traktowane jak zdarzenia aplikacji
    Set App = Application
End Sub
```

Przykład – formatowanie otwieranych skoroszytów

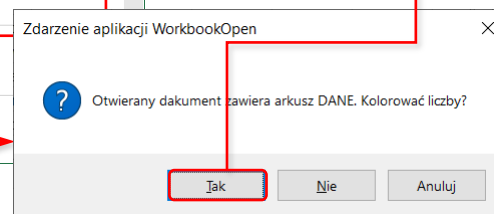
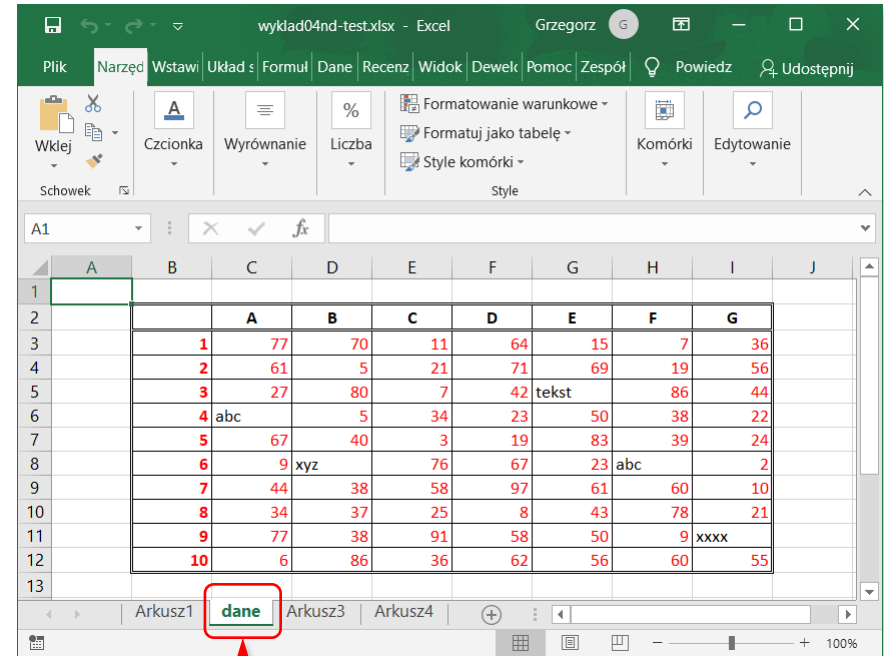
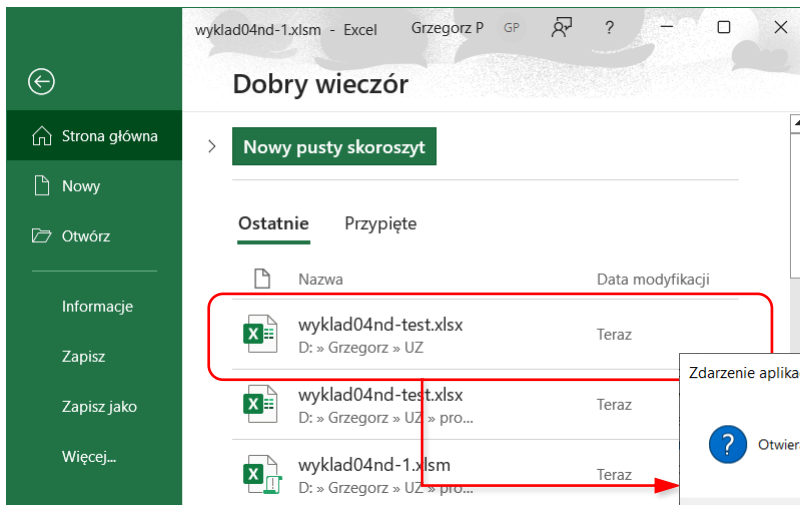
Makro jest uruchamiana przy otwieraniu dowolnego skoroszytu i sprawdza istniejące arkusze. Jeżeli znajdzie arkusz o nazwie „dane” formatuje wszystkie komórki zawierające wartości liczbowe ustawiając kolor tekstu na czerwony. Przed wykonaniem operacji formatowania wyświetla okno dialogowe z pytaniem o potwierdzenie.

```
Private Sub App_WorkbookOpen(ByVal Wb As Workbook)
    Dim sh As Worksheet
    For Each sh In Wb.Worksheets
        If (sh.Name = "dane") Then
            Dim odp As Integer
            Dim z As Range
            odp = MsgBox("Kolorować liczby?", vbYesNoCancel)
            If (odp = vbYes) Then
                Set z = sh.Cells.SpecialCells(xlCellTypeConstants, xlNumbers)
                z.Font.Color = vbRed
            End If
        End If
    Next
End Sub
```

Przykład – ustawienie wartości zmiennej

Zmienna obiektowa reprezentująca aplikację jest ustawiana przy otwieraniu skoroszytu zawierającego kod makra.

```
Private Sub Workbook_Open()  
...  
Set App = Application  
...  
End Sub
```



Dodatek (*add-in*) – dokument programu Excel zapisany jako plik „xlam”.

Cechy dodatków

- Są automatycznie ładowane podczas uruchomienia programu,
- Nie są widoczne jako dokumenty (nie należą do kolekcji Workbooks),
- Udostępniają swoje publiczne funkcje pozostałym dokumentom Excel-a,
- Ułatwiają dystrybucję kodu VBA.

Zarządzanie dodatkami

Plik->Opcje->Dodatki->Przejdź

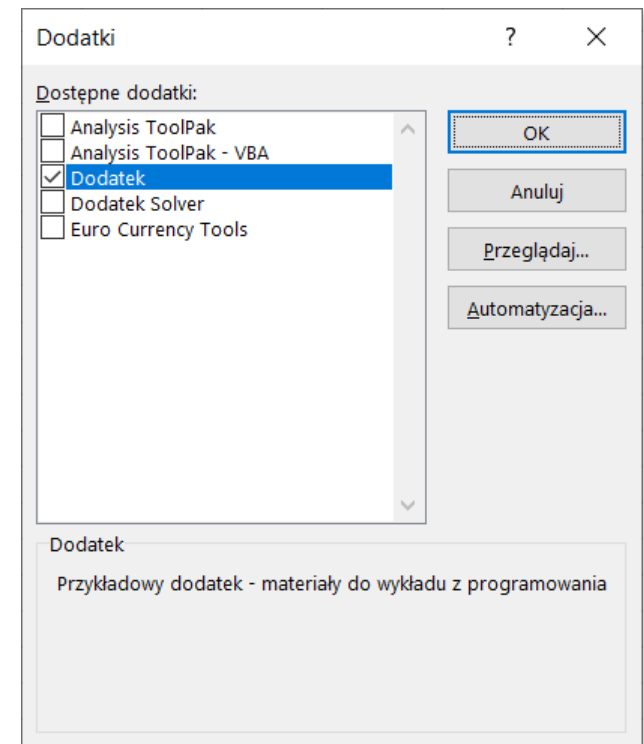
Developer->Dodatki programu Excel

Dołączenie nowego dodatku

Dodatki->Przeglądaj

Włączenie/wyłączenie dodatku

Pole opcji obok nazwy na liście dodatków



- ❑ Utworzenie nowego dokumentu xlsx, wprowadzenie makr, funkcji arkuszowych, formularzy i innych elementów, które mają być zawarte w dodatku.
- ❑ Testowanie wszystkich elementów.
- ❑ Jeżeli dodatek ma być chroniony hasłem w edytorze VBA wybrać:
Tools->VBAProject Properties,
na zakładce Protection zaznaczyć Look project for viewing i podać hasło.
- ❑ Opcjonalnie ustalić tytuł i komentarz (Plik->Informacje->Właściwości), będą wyświetlone w oknie Dodatki.
- ❑ Zapisać plik jako dodatek (Plik->Zapisz jako, typ pliku xlam), lokalizacja domyślna (może być zmieniona):
c:\Użytkownicy*użytkownik*\AppData\Roaming\Microsoft\AddIns
- ❑ Dodatek można przenosić pomiędzy komputerami i dodawać do programu (okno Dodatki, poprzedni slajd).

Na stronie przedmiotu dostępny jest dodatek (*add-in*) zapisany w pliku „dodatek.xlam”. Zawiera makra kolorujące komórki i wiersze, funkcję SumaLiczb (wykład 2.) oraz obsługę zdarzenia aplikacji WorkbookOpen (s.21). Przy ładowaniu tworzy menu z przyciskami uruchamiającymi kolorowanie. Menu jest usuwane przy zamykaniu/usuwaniu dodatku.

Przykład użycia dodatku zawiera arkusz wykład04nd-2.

```
Private Sub Workbook_Open()  
    Dim menu As CommandBar  
    Dim btn As CommandBarButton  
    Set menu = Application.CommandBars.Add("przykład", , False)  
    Set btn = menu.Controls.Add(msoControlButton)  
    btn.OnAction = "KolorujKomórki"  
    btn.Caption = "Koloruj komórki"  
    btn.Style = msoButtonCaption  
    ...  
    menu.Visible = True  
End Sub
```

