Tworzenie aplikacji graficznych SWT i JFace

Rozkład jazdy:

1. GUI przedpotopowe – era AWT i Swinga
2. Kilka słów na temat ewolucji SWT i JFace
3. “Standard Widget” – przegląd komponentów
4. Układy i układziki, czyli Layout Managery
5. Viewery i Wizzardy (dozorcy i czarodzieje!)

Łukasz Dywicki

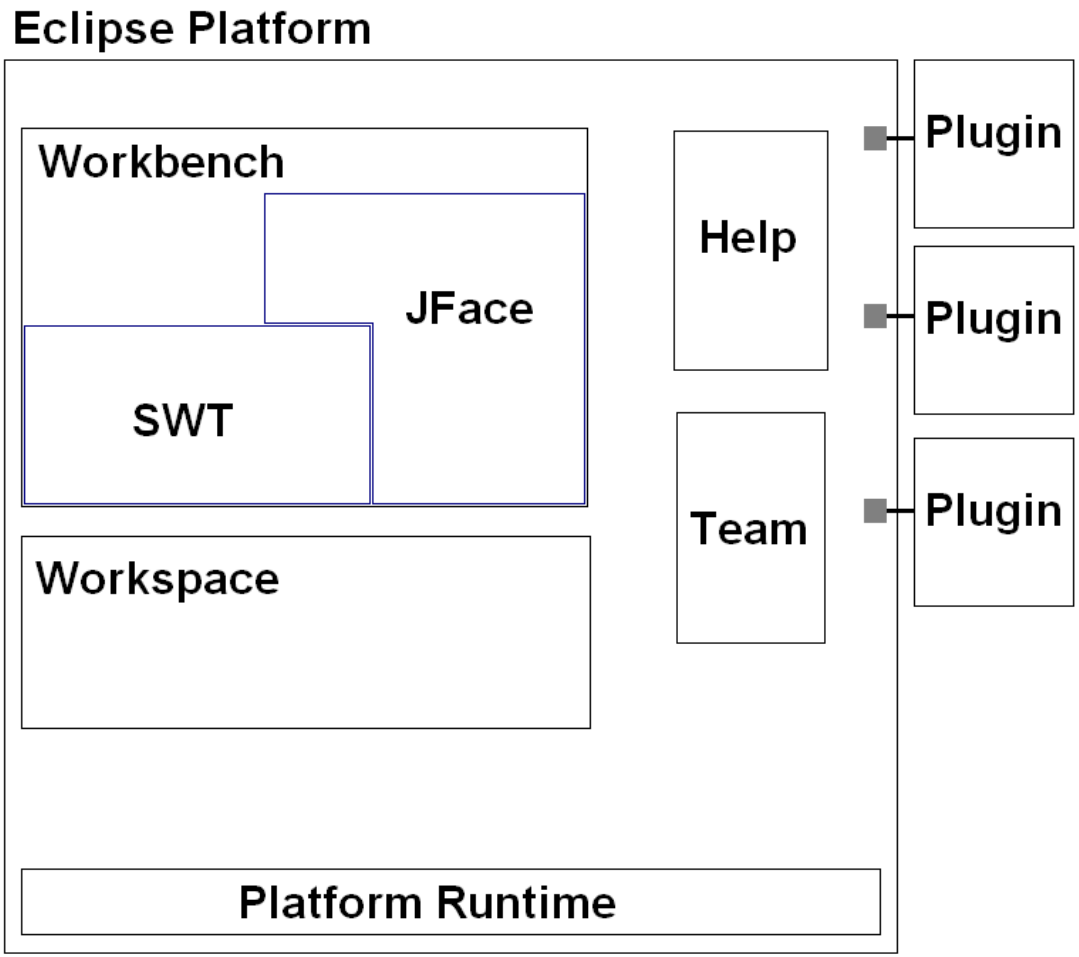
Tworzenie aplikacji graficznych SWT i JFace

1. Dawno, dawno temu istniała bestia LCD niszcząca ludzkie zbiory.
2. Pojawił się rycerz słońca i dał ludziom AWT.
3. Jednak AWT zbuntowało się przeciwko ludziom i zmieniło się w LCD.
4. Ponownie pojawił się rycerz słońca i obdarował ludzi Swingiem, który poskromił AWT.

Tworzenie aplikacji graficznych SWT i JFace

1. Za górami, za lasami był zakon 3 liter, który postanowił pomóc ludziom walczącym z AWT oraz sobie w walce ze złym księciem Małegosoftu.
2. Zakonnicy przywołali z zaświatów nowego stwora – SWT i jeźdźca JFace.
3. Ludzie się podzielili pomiędzy zwolenników potwora z zakonu i rycerza słońca.
4. Walka pomiędzy jednymi i drugimi trwa po dziś dzień.

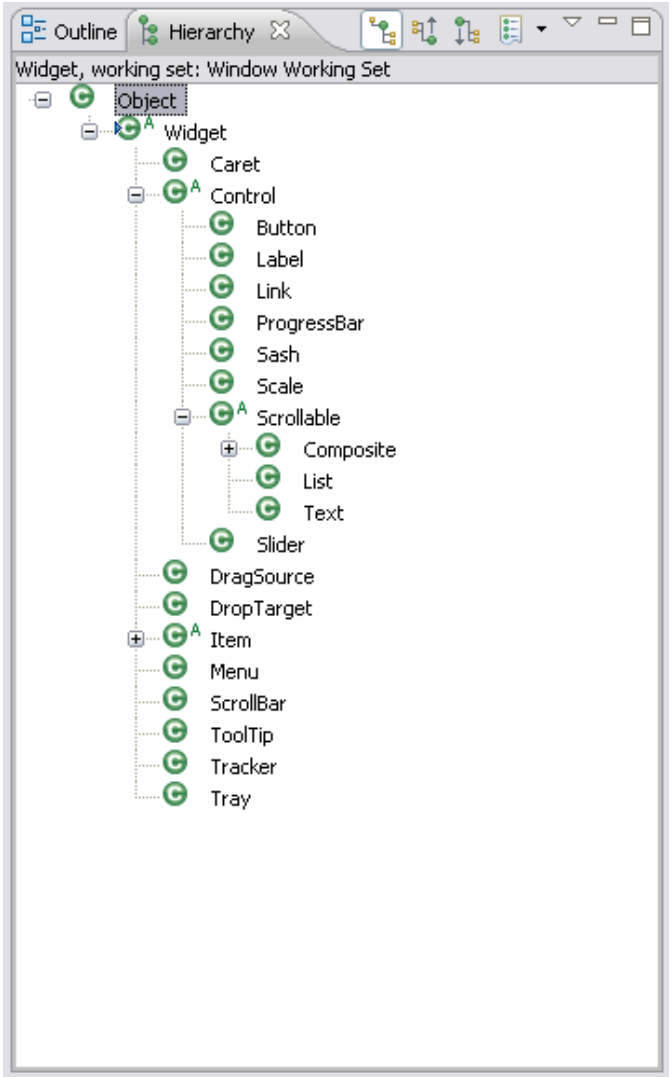
Tworzenie aplikacji graficznych SWT i JFace



Jest częścią platformy Eclipse

Może z powodzeniem być wykorzystywane samodzielnie w miejscu Swinga.

Tworzenie aplikacji graficznych SWT i JFace



Standard Widget Toolkit dostarcza kompletu komponentów prostych (np. przyciski) jak i złożonych (tabele, drzewa, listy).

Większość komponentów, które powołujemy do życia potrzebuje przodka.

Każdy komponent możemy stylizować przy pomocy stałych z klasy SWT. Większość komponentów

Tworzenie aplikacji graficznych SWT i JFace

Standardowe Layout Managery w SWT

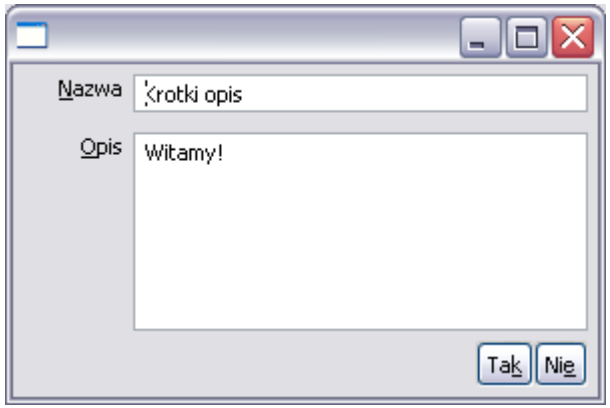
1. FillLayout – skaluje wszystkie komponenty do jednej wielkości
2. RowLayout – grupuje w kolumny bądź wiersze
3. GridLayout – rozmieszcza w siatce
4. FormLayout – układa komponenty absolutnie bądź relatywnie

Pierwszych dwóch używa się sporadycznie. Standardowa siatka jest najpopularniejsza ale i najtrudniejsza w modyfikacji.

Układ zarządzany przez FormLayout jest automatycznie skalowany. Ilość kodu, potrzebnego do pozycjonowania elementów jest w przypadku FormLayoutu o około 15% większa niż w przypadku siatki.

Tworzenie aplikacji graficznych SWT i JFace

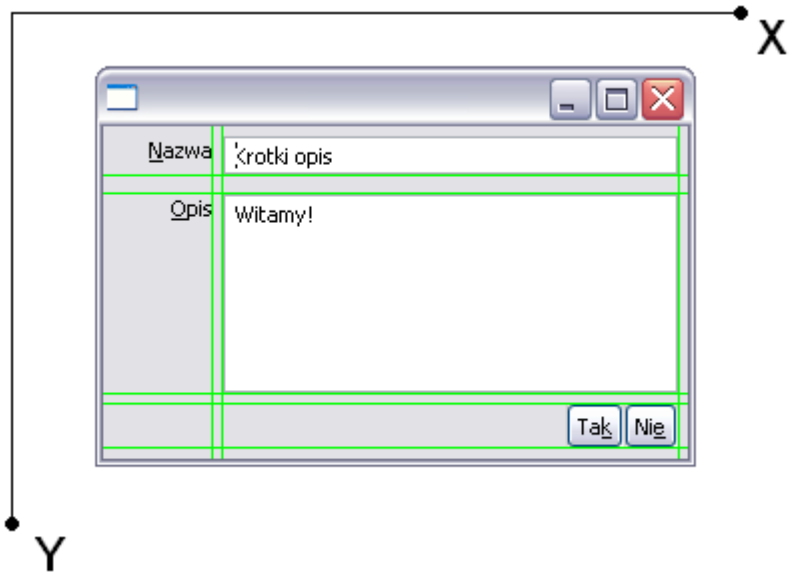
Rzut okiem na kod z siatki i formularza:




```
Text area = new Text(shell, SWT.MULTI | SWT.BORDER);
GridLayout layout = new GridLayout(3, false);
GridData data = new GridData();
data.horizontalSpan = 2;
data.heightHint = 90;
data.widthHint = 205;
area.setText("Witamy!");
area.setLayoutData(data);
```

```
Text area = new Text(shell, SWT.MULTI | SWT.BORDER);
FormLayout layout = new FormLayout();
FormData data = new FormData();
data.left = new FormAttachment(description); // etykieta tekstowa
data.right = new FormAttachment(100);
data.top = new FormAttachment(field, 5); // pole wiersz wyzej tekstowe
data.bottom = new FormAttachment(yes);  // przycisk pod polem
area.setText("Witamy!");
area.setLayoutData(data);
```

Tworzenie aplikacji graficznych SWT i JFace



Nieformalna siatka, którą stworzyliśmy przy użyciu obiektów FormData.

FormLayout skaluje elementy jeśli są one zadokowane w danej osi z obu stron.

FormAttachment ma zasadniczo dwa konstruktory. Pierwszy pozycjonuje relatywnie, drugi absolutnie.

```
new FormAttachment(description); // przypięcie danej strony do określonego komponentu
new FormAttachment(100); // rozciągnięcie do 100% w danym kierunku
```

Osie X oraz Y wskazują kierunki, względem których rosną wartości.

Tworzenie aplikacji graficznych SWT i JFace

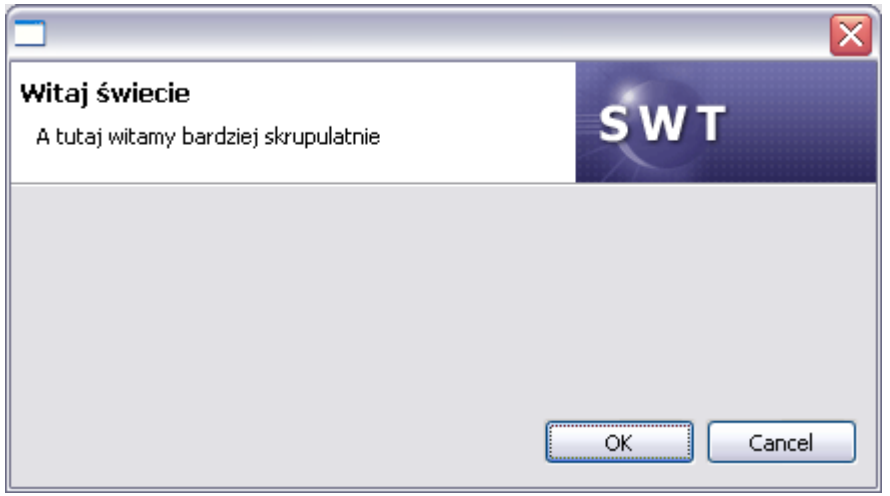
JFace to nadbudowa na SWT pozwalająca zmniejszyć liczbę kodu potrzebnego do uzyskania zamierzonego efektu.
Jest to również pewnego rodzaju “Swing” dla SWT dostarczający implementacji MVC.
Najczęściej używane komponenty to:

1. Okna dialogowe
2. JFace Viewer's
3. Akcje
4. Wizardy

Tworzenie aplikacji graficznych SWT i JFace

Standardowy układ okna wraz z paskami narzędziowymi/statusem, menu itp. możemy stworzyć przy pomocy ApplicationWindow. Informujemy o tym w kontruktorze metodami:

1. addToolBar
2. addCoolBar
3. addMenu
4. addStatusLine



ApplicationWindow po wywołaniu tych metod odwołuje się do domyślnych implementacji. Zmieniając je możemy dostosowywać okno do własnych potrzeb.
Do dyspozycji mamy również standardowy zestaw dialogów.

Tworzenie aplikacji graficznych SWT i JFace

Każdy Viewer wymaga dostarczenia implementacji 2 interfejsów:

1. IContentProvider – dostarcza obiektów, które mają być wyświetlone
2. ILabelProvider – określa etykiety tekstowe oraz ikony prezentowane przy obiektach zwróconych przez content providera

Opcjonalnie możemy dostarczyć implementacji dwóch kolejnych interfejsów:

1. IViewSorter – sortowanie elementów, które są wyświetlane
2. IViewFilter – ograniczenie ilości wyświetlanych elementów

Tworzenie aplikacji graficznych SWT i JFace

Stworzenie prostego TreeViewera:

```
TreeViewer tree = new TreeViewer(form, SWT.SINGLE);
tree.setLabelProvider(new NavigationTreeLabelProvider());
tree.setContentProvider(new FilesystemContentProvider());
tree.setInput(new File("."));

public class NavigationTreeLabelProvider extends LabelProvider{
    public String getText(Object element) {
            return ((File) element).getName();
    }
}
```

Pozostałe Viewery tworzy się i konfiguruje w ten sam sposób.

Istnieją różne wariacje tego komponentu np. TableTreeViewer, CheckboxTreeViewer. Podobnie sprawa ma się z ListViewerem.

Tworzenie aplikacji graficznych SWT i JFace

Akcje to wydzielone fragmenty kodu, które mogą być wielokrotnie używane w celu podjęcia tej samej czynności w różnych miejscach.

Akcja w rozumieniu JFace to implementacja interfejsu IAction. Jak w większości przypadków mamy domyślną implementację większości metod w klasie Action.

Akcję możemy opisać tekstem bądź udekorować obrazkiem.

Stworzona w ten sposób akcja może być używa w ten sam sposób przez MenuManagera (do budowania menu) czy też ToolBarManagera (do budowania paska narzędziowego).

Tworzenie aplikacji graficznych SWT i JFace

Przy użyciu kreatorów możemy tworzyć długie “klikalne” sekwencje pytań.
Implementacja nowego kreatora wymaga od nas implementacji interfejsów IWizard oraz IWizardPage.
Pierwszy reprezentuje kreator, drugi pojedyńczy krok. Są domyślne implementacje Wizard oraz WizardPage.
Każda strona może przeprowadzić walidację i decydować o tym czy proces może być zakończony. Domyślnie Wizard sprawdza czy wszystkie strony wyraziły zgodę na zakończenie prac.

Rozbudowa kreatorów przebiega tak jak w przypadku ApplicationWindow – nadpisujemy puste implementacje metod.

Tworzenie aplikacji graficznych SWT i JFace

Jako bonus – drag & drop.
Jest zorganizowany podobnie jak w Swingu – możemy obsługiwać różne formaty danych wynikowych.
Kluczowe znaczenie mają dwie klasy

1. DragSource – miejsce, z którego będziemy przenosić
2. DropTarget – miejsce, na które ma trafić dany element

Wpływ na przebieg procesu mamy przez dodawanie odpowiednich listenerów.
Akceptowane typy danych docelowych określamy przez obiekty klasy Transfer (FileTransfer, RTFTransfer itp.).

Tworzenie aplikacji graficznych SWT i JFace

Kilka słów na temat wad SWT i JFace

1. Stosunkowo mało dokumentacji (Swing ma więcej), tylko “kilka” książek.
2. Brak rozbudowanej społeczności (Swing ma więcej), ogranicza się do list dyskusyjnych.
3. Brak polskojęzycznych serwisów (ale tu ogólnie o Javie w polskiej sieci publikuje się mało)
4. Mało ofert pracy (chociaż są nieco lepiej płatne – na podstawie itjobswatch.co.uk)

Tworzenie aplikacji graficznych SWT i JFace

Dostępne pozycje książkowe to:

1. The Definitive Guide to SWT and JFACE (Apress)
2. SWT/JFace in Action (Maning)
3. Proffesional Java Native Interface with SWT/JFace (Wrox)
4. SWT: The Standard Widget Toolkit volume 1 (Addison-Wesley)
5. SWT: A developer's notebook (O'Reilly)

Tworzenie aplikacji graficznych SWT i JFace

Może zanim się rozejdziemy mały flame?

1. Dlaczego nie tworzymy aplikacji desktopowych?
2. Czy jest to lepsze niż HTML?
3. Jakie problemy mogą powstać przy tradycyjnych desktopach?

System.exit(0);