03. Rozszerzenia przykładowe i testowe

W tym rozdziale opisane są rozszerzenia, które powstały w celu przetestowania mechanizmów rozszerzeń w aplikacji redaktora i administratora oraz mogą służyć jako przykłady implementacji wtyczek dla programistów zainteresowanych tym tematem.

Budowanie rozszerzeń

> **Kod źródłowy**
>
> Kod źródłowy przykładowych rozszerzeń można pobrać tutaj: dcore-app-extension-tests.zip

Aby było możliwe zbudowanie przykładowych rozszerzeń z ich plików źródłowych, w systemie musi być zainstalowane narzędzie Maven 1.1.

Po rozpakowaniu archiwum z kodem źródłowym, w katalogu `dcore-app-extension-tests` znajdują się katalogi projektów dla poszczególnych rozszerzeń, o nazwach zaczynających się od `dcore-app-extension`. Znajduje się tam też katalog `maven-repo`, który zawiera dodatkowe pliki `jar` potrzebne przy budowaniu.

Aby zbudować wybrane rozszerzenie, należy w konsoli wejść do powiązanego z nim katalogu i wywołać polecenie

```
maven clean install
```

Można również zbudować wszystkie rozszerzenia na raz. W tym celu w katalogu `dcore-app-extension-tests` należy wywołać polecenie

```
maven multiproject:clean multiproject:install
```

Dla programistów pracujących w środowisku Eclipse, przydatne będzie polecenie, które tworzy w katalogu wtyczki projekt Eclipse gotowy do zaimportowania:

```
maven eclipse
```

Po zbudowaniu danego rozszerzenia w katalogu `target` zostaje umieszczony plik `jar`, który można zainstalować w dLibrze.

Instalacja rozszerzeń

Przykładowe rozszerzenia nie są domyślnie zainstalowane w Aplikacji Redaktora i Administratora dLibry. Aby zainstalować wtyczkę w systemie, należy skopiować plik `jar` wtyczki do katalogu /WEB-INF/jnlp-jars aplikacji czytelnika i uruchomić w panelu administracyjnym proces uaktualnienia aplikacji redaktora/administratora. Więcej na temat panelu administracyjnego aplikacji czytelnika można dowiedzieć się tutaj.

Po ponownym uruchomieniu Aplikacji Redaktora wtyczki powinny być widoczne (można to sprawdzić w menu Pomoc -> O programie). Proces uaktualnienia w panelu administracyjnym musi zostać uruchomiony po każdej zmianie plików `jar`.

Konfiguracja rozszerzeń

Aby rozszerzenie było poprawnie widoczne w Aplikacji Redaktora i Administratora, odpowiednia konfiguracja musi znajdować się w pliku `src/etc/plugin.xml`. Budowa tego pliku wynika z wymagań biblioteki Java Plugin Framework. Poniżej znajduje się przykładowa zawartość tego pliku:

plugin.xml

```
<?xml version="1.0" ?>
<!DOCTYPE plugin PUBLIC "-//JPF//Java Plug-in Manifest 0.7" "http://jpf.sourceforge.net/plugin_0_7.dtd">
<plugin id="pl.psnc.dlibra.app.extension.op.test" version="${pom.currentVersion}">
<requires>
        <import plugin-id="pl.psnc.dlibra.app.extension"/>
</requires>
        <runtime>
#foreach($dep in ${pom.dependencies})
#if(${dep.type} == "jar" && !${dep.getProperty('dist.skip').equals("true")})
                <library id="${dep.artifact}" path="lib/${dep.artifact}" type="code" />
#end
#end
                <library id="pluginCode" path="${jarName}" type="code" />
        </runtime>
<extension plugin-id="pl.psnc.dlibra.app.extension"
        point-id="objectPanel" id="op-test">
        <doc>
                <doc-text>Polish interface language is provided by Poznan Supercomputing and Networking Center.<
/doc-text>
        </doc>
        <parameter id="class"
                value="pl.psnc.dlibra.app.extension.optest.DummyObjectPanel"/>
</extension>
</plugin>
```

W znaczniku `<runtime>` znajduje się makro Velocity, które automatycznie dołączy do dystrybucji wtyczki niezbędne pliki `jar` (wszystkie te, które są wymienione jako zależności w pliku `project.xml` i nie mają dopisanej właściwości `<dist.skip>true</dist.skip>`).

Znacznik `<extension>` definiuje konkretne rozszerzenie. Atrybut `pluginId` musi mieć wartość `"pl.psnc.dlibra.app.extension"`. Atrybut `point-id` musi zawierać identyfikator punktu rozszerzeń, do którego ma być włączona wtyczka (zgodny z nazwami wymienionymi w dokumentacji), natomiast atrybut `id` musi być unikalnym identyfikatorem rozszerzenia. Wewnątrz znacznika `<extension>` można umieścić opcjonalny znacznik `doc` z tekstem, który ma być wyświetlony w oknie informacyjnym Aplikacji (Menu Pomoc -> O programie...), oraz znaczniki `<parameter>` definiujące parametry wymienione w opisie danego punktu rozszerzeń.

Uwaga: w przykładowych rozszerzeniach każda wtyczka korzysta z tylko jednego punktu rozszerzeń, ale nic nie stoi na przeszkodzie, aby jedna wtyczka była włączona do wielu punktów. W tym celu należy umieścić w pliku konfiguracyjnym więcej znaczników `<extension>`.

Lista przykładowych rozszerzeń

Rozszerzenia zostały wymienione i opisane w podrozdziałach:

- 01. Przykładowe rozszerzenie typu tool
- 02. Przykładowe rozszerzenie typu eventListener
- 03. Przykładowe rozszerzenie typu objectPanel
- 04. Przykładowe rozszerzenie typu dataSource