



PEtALS Quick Start

This document presents the Quick Start release of PEtALS. This release targets PEtALS beginners to ease their first step with PEtALS.

PEtALS Team

Roland NAUDIN <roland.naudin@ebmwebsourcing.com>

- February 2008 -



(CC) EBM WebSourcing - This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc-sa/3.0/>



Table of Contents

Preface	4
1. Presentation	5
1.1. What is PEtALS?	5
1.2. Quick Start features	5
1.3. Directories structures	6
1.4. Acronyms	6
2. PEtALS container	7
2.1. Pre-Requisites	7
2.2. Installation	7
2.3. Directory structure	7
2.4. Configuration	7
2.5. Starting PEtALS	8
2.6. Stopping PEtALS	8
3. Use cases	9
3.1. The Clock use case	9
3.2. The Helloworld use case	9
3.3. The File Transfer use case	10
3.4. The Transformation use case	11
4. Next steps... ..	13

List of Figures

1.1. PEtALS is an ESB fully JBI compliant	5
3.1. File Transfer use case	10
3.2. Transformation use case	12

Preface

PEtALS delivers the OW2 Java Business Integration (JBI) bus. See [JSR-208 specifications](#) for further details on JBI.

Since its beginning, PEtALS has increased its coverage amongst different high value domains; clustering, robustness, availability, performance... Moreover, PEtALS relies entirely on its OW2 partner technology, the Fractal component model, which brings to its architecture a strong modularity. Please visit the Fractal web site for further details at <http://fractal.objectweb.org>.

At the version 2.1 of the PEtALS kernel, the PEtALS team have decided to exploit Fractal leverage by delivering various PEtALS distributions. Each distribution is packaged and customized to be addressed to a specific audience.

As its name suggests, this distribution is attended to PEtALS beginners, it proposes an all-in-one PEtALS container with an embedded Web console, simple use cases and this getting-started documentation.

The documentation leads you through the PEtALS universe and introduces the specialized documentations when needed.

Chapter 1. Presentation

1.1. What is PEtALS?

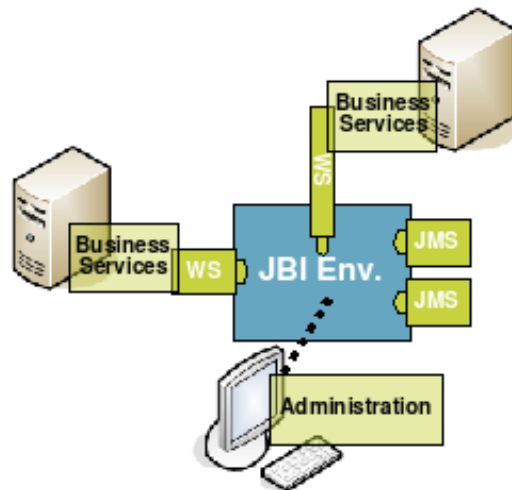
PEtALS helps you to integrate your Enterprise Business Units in order to provide a coherent and rational global solution. With PEtALS, you can compose your new applications by integrated already existing ones and new ones.

Thus, all your applications expose their business logic by exposing services. This concept is well known as Service Oriented Architecture (SOA).

PEtALS offers a solid backbone for your enterprise Information System and acts as a Bus, a place where all your data are exchanged. PEtALS connects services to each others. Petals is an Enterprise Service Bus (ESB).

Please visit our Web site for further details <http://petals.ow2.org>.

Figure 1.1. PEtALS is an ESB fully JBI compliant



1.2. Quick Start features

The Quick Start package contains the following elements:

- **Standalone PEtALS container** : A standalone version of PEtALS container, with an unique local transporter which transfers Message Exchange in memory (as reference). This version has been lightened of reliability, availability and clustering features.
- **Embedded Web Console** : The PEtALS Web Console embedded into the PEtALS container, which provides an enhanced GUI to monitor and administrate the PEtALS container.
- **Sample Components** : A bunch of component samples helpful to make any quick tests on the PEtALS Bus.
 - *petals-sample-client* : A graphical client to process quick visual tests based on JBI component API.
 - *petals-sample-clock* : A "ready at deployment" component which exposes a clock service automatically at its starting, without using the SA (Service Assembly) deployment mechanism.
 - *petals-sample-helloworld* : A simple component which exposes a helloworld service at the deployment of a basic SU (Service Unit) contained in a SA.
- **Components** : A selection of components used along the Quick Start use cases.
 - *petals-bc-filetransfer* : A BC (Binding Component) which permits to transfer files to and from the PEtALS Bus.

- *petals-bc-soap* : A BC supporting Web Services. This component provides most of the features of WS-*, but we introduce only a simple usage in the Quick Start use cases.
- *petals-se-xslt* : A SE (Service Engine) which provides XSLT transformation features.
- *petals-se-eip* : A SE bringing a library of integration patterns, usefull to provide light orchestration. We use only a simple pattern in the Quick Start use cases.
- **Use Cases** : 4 use cases, fully documented, which guide you accross the PEtALS universe. Each use case introduce a new element to let you assimilate each important notions.
 - The *Clock* use case : This first use case makes you uses the Clock sample component and the Client sample component to invoke your first service in PEtALS.
 - The *Helloworld* use case : This use case lets you use the Web console to deploy the Helloworld sample component and its associated SA. The service invocations are still done with the Client sample component.
 - The *FileTransfer* use case : This use case introduces the usage of JBI dedicated ant tasks (<http://ant.apache.org>) and make you process file transfers with PEtALS.
 - The *Transformation* use case : This final use case introduces a more complex integration with a mini orchestration. It relies on the components SOAP, EIP and XSLT.

Once achieved the 4 use cases, you will be able to build your own integrations; you 'll be a new member PEtALS users community!

1.3. Directories structures

The Quick Start distribution is organised in 4 main directories:

- **ant-sample** : includes a sample file containing exemples to illustrate the use of JBI specialized ant tasks.
- **PEtALS** : Contains the PEtALS Quick Start container with its embedded Web Console.
- **sample-components** : Contains the samples components required to run the Quick Start use cases.
- **components** : Contains the components required to run the Quick Start use cases.
- **usecases** : Contains the usecases *Helloworld*, *FileTransfer* and *Transformation*.

1.4. Acronyms

Along the document, we are using common JBI acronyms :

- **SL** : a Shared Libraries artifact.
- **BC** : a Binding Component artifact.
- **SE** : a Service Engine component artifact.
- **SA** : a Service Assembly artifact.
- **SU** : a Service Unit artifact.
- **MEP** : Message Exchange Pattern, JBI defines 4 possible MEP to invoke a service.

Chapter 2. PEtALS container

2.1. Pre-Requisites

- **Java** : To run PEtALS, you need at least a Java JRE 1.5 distribution.

The Sun JVM can be downloaded at : http://java.sun.com/javase/downloads/index_jdk5.jsp.

- **Ant** : To run the provided use cases, you need to install an Ant distribution.

Apache Ant can be downloaded at : <http://ant.apache.org/bindownload.cgi>.

2.2. Installation

PEtALS doesn't require any installer to be installed. Just copy the provided directory *PEtALS* anywhere you want to install a Quick Start PEtALS.

Generally, PEtALS find by itself its directory location. On some systems, the installation path can not be automatically found, in such case you must set the *PETALS_HOME* environment variable :

- On Unix like system : `export PETALS_HOME=your_installation_path.`
- On Windows like system : `set PETALS_HOME=your_installation_path.`

2.3. Directory structure

- `bin` : includes scripts to launch PEtALS on Unix (*.sh) or Windows (*.bat) systems.
- `conf` : includes PEtALS configuration files.
- `install` : An auto-loader directory; put a JBI component (SE or BC) or a SA to have them automatically installed, deployed and started in the PEtALS container.
- `installed` : components and services assemblies are automatically copied into this directory after a succesful installation (or deployment).
- `lib` : includes all libraries required by PEtALS system.
- `logs` : includes all logs generated during PEtALS execution.
- `lost+found` : contains lost JBI elements (components or SAs no more referred in the PEtALS repository).
- `repository` : includes all the libraries, config files (etc) of the installed or deployed components, SLs and SAs.
- `schema` : includes some usefull XML schemas used by PEtALS.
- `uninstalled` : components, SLs and SAs are automatically moved into this directory after a succesful uninstallation (or undeployment).
- `webapps` : Contains the Web console package to be deployed into Quick Start PEtALS.
- `work` : used by PEtALS system to put its intern resources during runtime.

2.4. Configuration

The Quick Start PEtALS is pre-configured to work properly.

If you want to customize the configuration, please use the PEtALS Standalone distribution and the PEtALS Platform distribution.

2.5. Starting PEtALS

1. Go to `$PETALS_HOME/bin`
2. Launch `startup.sh` or `startup.bat` :

```
# ./startup.sh
```

You can launch PEtALS in console mode, to be able to interact with it via the terminal. In this case, use the option `-C` in the terminal :

```
# ./startup.sh -C
```



Note

You can also use directly the `jar server.jar` in the `$PETALS_HOME/bin`, with the command `start`, `stop`, `shutdown` as argument.

2.6. Stopping PEtALS

There is several ways to stop PEtALS :

1. If PEtALS is launched in a simple terminal, type `<ctrl>+ c` to stop PEtALS.
2. If PEtALS is launched in console mode, just type `q` or `x` and `<enter>` in the terminal :

```
# q
```

By typing `q`, PEtALS is stopped and component, SA, SL aren't uninstalled/undeployed.

By typing `x`, PEtALS is shut down; component, SA, SL are uninstalled/undeployed.

3. If PEtALS is launched in background, launch the `stop.sh` script :

```
# ./stop.sh
```

4. If PEtALS is launched in background, launch the `shutdown.sh` script :

```
# ./shutdown.sh
```


Chapter 3. Use cases

3.1. The Clock use case

With this simple use case, you will be able with few manipulations to invoke your first service in the PEtALS Bus.

Here are the steps to follow :

- Start your PEtALS engine (see [section 2.5](#)) in console mode
- Copy the Clock sample component located in the directory `sample-components` into the directory `$PETALS_HOME/install`. The PEtALS auto-installer install and start automatically the component
- In the PEtALS terminal, type `d`, the terminal will display the content of the PEtALS JNDI repository:

```
JNDI directory local
<== / ==>

    <=users=>

    <=new-endpoints=>
        -> {http://petals.ow2.org}ClockService@ClockEndpoint [Link to
/endpoints/{http://petals.ow2.org}ClockService@ClockEndpoint]

    <=services=>

        <={http://petals.ow2.org}ClockService=>
            -> {http://petals.ow2.org}ClockService@ClockEndpoint [Link to
/endpoints/{http://petals.ow2.org}ClockService@ClockEndpoint]

    <=endpoints=>
        -> {http://petals.ow2.org}ClockService ->ClockEndpoint
(INTERNAL):subdomain1/0/petals-sample-clock

    <=interfaces=>
```

You can see the new endpoint exposed by the Clock component when it's starting.

- Copy the Client sample component located in the directory `sample-components` into the directory `$PETALS_HOME/install`. A graphical interface is displayed at the start of the client component.
- With the help of the Client component documentation (<http://petals.objectweb.org/components.html>), try to invoke the `printTime` operation to the `ClockEndpoint`.
- Once your first service invocation done, try to change the MEP (Message Exchange Pattern) and the possible operations with the help of the service description (WSDL); see the error or fault generated when the operation or pattern is not supported by the Clock component.
- To clean your PEtALS server, type `x` on the PEtALS console. All the installed JBI artefacts are automatically uninstalled and the PEtALS container is shut down.

3.2. The Helloworld use case

This use case introduces the usage of the PEtALS Web console. This console allows you to administrate and monitor you PEtALS container in a nice and intuitive way.

Follow the steps:

- Start your PEtALS engine (see [section 2.5](#))

- With your favorite Web browser, connect you to you Web console at the url <http://localhost:7878>. Use default values to configure the Data Collector and to log in.
- With the help of the Web Console documentation (<http://petals.objectweb.org/documentation.html>), install and start the Helloworld sample component and the Client sample component, located in the directory `$PETALS_HOME/install`.
- Once the components installed, unzip the use case Helloworld, located at `usecases/petals-simplehelloworld.zip` in the Quick Start package. Deploy and start the contained SA named `sa-simplehelloworld.zip` by the Web console.
- If you unzip the `sa-simplehelloworld.zip` SA, you will see that it contains another zip file, named `su-simplehelloworld-provide.zip`. This zip file is in fact a SU which describes a provide service for the Helloworld component. Check in each JBI artifact the file `META-INF/jbi.xml`. This file is the JBI descriptor, the standard element which describes all the relevant information to build service configurations in JBI components.



Note

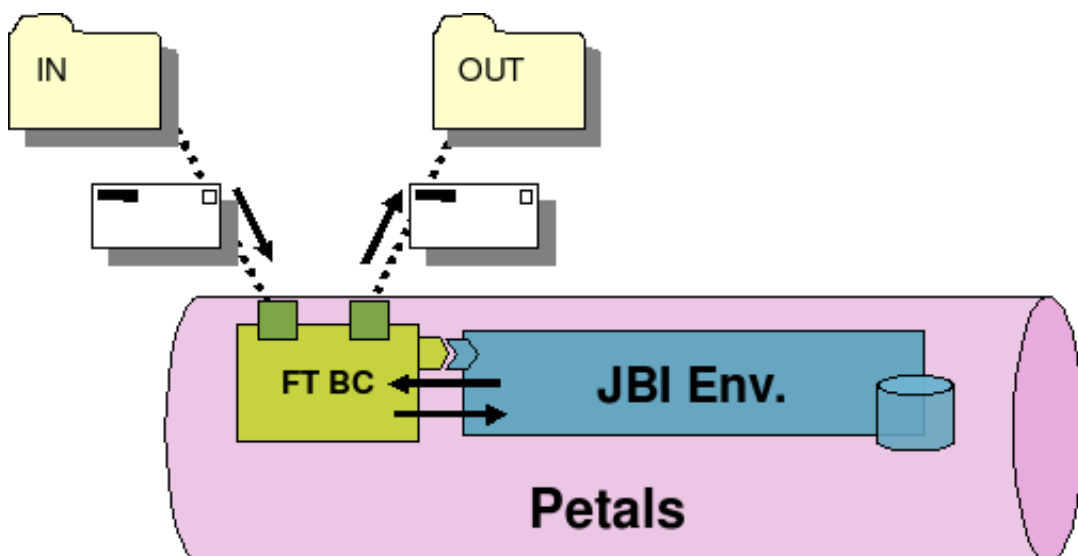
If you check in the JBI components zip files, you will find the same JBI descriptor. In this case, the JBI descriptor defines the component configuration itself. If you want more information on these descriptors, please see the JBI specifications.

- With the Client sample GUI, get the service description (WSDL) of the Helloworld service and retrieve the available operations. Invoke the different offered operation like for the previous use case and see what happened.
- To clean your PEtALS server, type `x` on the PEtALS console. All the installed JBI artefacts are automatically uninstalled and the PEtALS container is shut down. You cant just shut down PETALS by typing `q` for example. When you will restart PETALS, the installed JBI elements would be recovered to retrieve their previous life-cycle states.

3.3. The File Transfer use case

Now that you are more familiar with PEtALS and the sample components, we will introduce a real case of integration. This integration is very simple, it consists to poll files from a local directory named `in` and transfer it to another directory named `out`. The file is conveyed through the PEtALS BUS in a Message Exchange form.

Figure 3.1. File Transfer use case



In this use case, we introduce the usage of ant technology and specially the JBI dedicated ant tasks provided by PEtALS.



Note

You must have `Ant` installed on your system to process the use case. (see [section 2.1](#)).

Please follow the steps:

- Unzip the File Transfer use case located at `usecases/petals-simplefiletransfer.zip`.
- In the use case exploded directory, you can find 3 directories and a file:
 - **deployables** : This directory contains all the JBI artifacts to install or deploy into PEtALS to run the use case.
 - **lib** : This directory contains the libraries required to process PEtALS ant tasks.
 - **test** : This directory contains the resources required to process a test of the deployed use case.
 - **build.xml** : This XML file contains the definition of the ant goals used to run the use case.
- For this use case, you need to specify in your working environment the variable `PETALS_HOME` (see [section 2.2](#)).
- Start your PEtALS engine.
- To process the use case, invoke the ant goals in the following order from the use case root directory:
 - *prepare* : This goal prepares the environment to run the use case, it creates directories in `$PETALS_HOME`.

```
# ant prepare
```

- *deploy* : This goal installs, deploys and starts all the JBI artefacts required to run the use case.

```
# ant deploy
```

- *run* : This goal runs the test of the use case. It copies a file in the directory `$PETALS_HOME/usecases/simplefiletransfer/in` and wait that a file named `gettingstarted.xml` arrived in the directory `$PETALS_HOME/usecases/simplefiletransfer/out`, which represents the end of the integration processing.

```
# ant run
```

- *clean* : This goal cleans the PEtALS environment, it removes the deployed components, SA, and removes created directories.

```
# ant clean
```

You can have a look to the petals-bc-filetransfer component documentation (<http://petals.objectweb.org/components.html>) for further details about the configuration of SUs for the component.

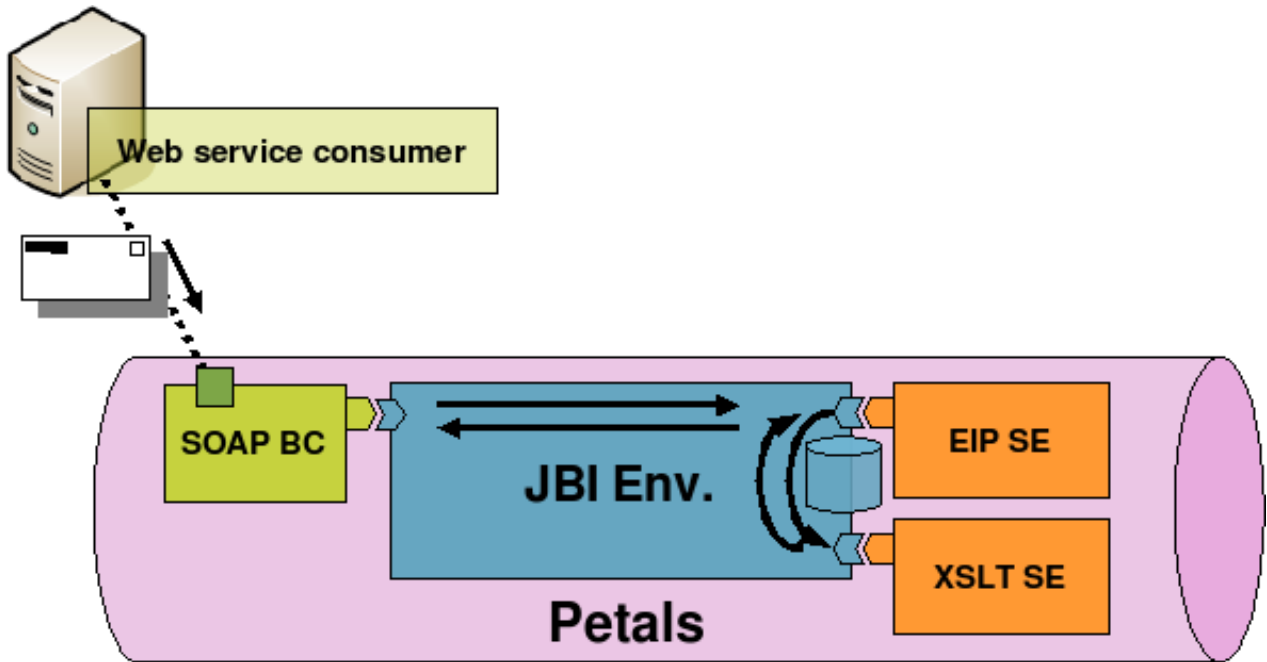
You can have a look to the samples of ant task in the directory `ant-sample` of the Quick Start package, to have an illustration of all the possible PEtALS ant tasks.

3.4. The Transformation use case

This last use case provides a more complex integration than an IN/OUT message flow. It involves the orchestration of Message Exchange to build the complex message flows.

In this use case, we introduce the usage of the `EIP` component to build a basic orchestration. The ingoing flow is exposed by a Web Service via the `SOAP` component, and the service itself is a transformation processed by the `XSLT` component.

Figure 3.2. Transformation use case



Here are the steps to follow:

- Unzip the Transformation use case located at `usecases/petals-simpletransformation.zip`.
- Start your PETALS engine if you have stopped it.
- Invoke the ant goals in the following order from the use case root directory:
 - *deploy* : This goal installs, deploys and starts all the JBI artefacts required to run the use case.

```
# ant deploy
```

- *run* : This goal runs a java program. This program is a Web Service client which processes a request to the transformation Web Service and controls if the returned document is correctly formatted.

```
# ant run
```

- *clean* : This goal cleans the PETALS environment, it removes the deployed components and SA.

```
# ant clean
```

Don't hesitate to pick a glance to the SA package in the `deployables` directory of the use case. You can see how the SUs are configured compared to the related component documentations (<http://petals.objectweb.org/documentation.html>).



Note

If you want to test this use case with a graphical Web Service client, please have a look to the Soap use cases documentation to get information about how using SOAPUI product.

Chapter 4. Next steps...

Now that you are familiar with PEtALS, you can start creating your own integration or service oriented application based on the large stock of PEtALS components. Check the available components at <http://petals.objectweb.org/components.html>.

If you want to control or optimize the PEtALS configuration, please refer to the Standalone distribution documentation.

If you are interesting about running PEtALS in a distributed environment, please refer to the Platform distribution documentation.

You can be interested to build your own components and thus contribute the PEtALS community. PEtALS offers a framework which handles most of the parts of JBI mechanisms and lets you focus on your business logic: the CDK.

PEtALS team works with proved quality tools to develop PEtALS. The team has written a guide to lead any interested developer who wish to join the PEtALS developer team (<http://petals.objectweb.org/documentation.html>).