icw
eHEALTH FRAMEWORK

eHealth Framework

Creating a new Web Service

Imprint

InterComponentWare
Altrottstraße 31
69190 Walldorf
Tel.: +49 (0) 6227 385 0
Fax.: +49 (0) 6227 385 199

Document version: preliminary
Document Language: en (US)
Product Name: eHealth Framework
Product Version: 2.10.3
Last Change: 23.03.2010
Editorial Staff: BAS Technology

Notice

The wording in this document applies equally to women and men. The masculine form was selected to ease the comprehensibility and legibility of the text.

Note:

The current version of this document has a draft status and various chapters are still in review.

The document is collaboratively built with the use of the Darwin-Information-Typing-Architecture (DITA) and has therefore a draft status concerning styles and layout. The necessary adaptations are currently also in a developmental stage.

Contents

1 Overview

Purpose

The purpose of this howto is to provide a step-by-step guide of how to create a new web service. This document leads the reader beginning with an empty directory all the way towards packaging and distributing the new artifacts.

Scope

The howto describes how a new custom web service can be created. It does not include the description on how a module can be added to an application. This is subject to a separate howto.

For more details on the eHealth Framework please refer to the eHealth Framework Reference Documentation on page 5.

2 Creating a new Web Service

2.1 Creating a new Service Module Project

In order to get started the eHF module template is required. It is contained in the ehf-project-templates project, which is part of the eHF distribution.

The goal of this task is to create the skeleton of a new eHF-based module, using the eHF module template and maven. The resulting project can be easily imported into an Eclipse-based IDE.

1. Open a console window

   On windows operating systems this can be accomplished using the classic start menu and selecting the `Run...` menu item. In the dialog you simply have to type `cmd` and a Windows console will be presented.

   On unix-based operating systems you open a new shell or console.

2. Change into your workspace directory

   On windows operating systems this can be accomplished by first changing to the drive the workspace is located. E.g. this is the C drive you type `c:`. Then you simply use the `cd <foldername>` command substituting the `foldername` with the name of the folder or a complete path until you are inside the workspace folder.

   On unix-based operating systems you normally do not have to deal with different drives. They should have been mounted using an appropriated and known alias.

3. Create the project structure using maven genapp
   Execute the following command in the workspace directory:

   ```
   maven genapp
     -Dmaven.genapp.template.dir=<project-templates>/ehf-module-template
   ```

   (where `<project-templates>` is replaced with the path to the ehf-project-templates project, that you obtained in the previous step – this path can either be expressed as a relative or absolute path.)

   Maven prompts for further information.

4. Please specify the project root directory
   The root directory is the directory the project will be created in. The default is not useful in this case as it points directly to your workspace. Specify the name of the folder here that should be used. The folder can be relative and must i.e. not provide an absolute path.

5. Please specify an id for your project
   The id has many purposes. It will be the `id` of the project as well as the artifact id for distribution.

6. Please specify a groupId for your application
   The `groupId` is propagated in the project description. It is also used to define the group the final artifact will go to in the maven repository.

7. Please specify a name for your project
   This is for providing a display name of your project. The information will be propagated into the `project.xml` project description file.

8. Please specify the package for your application

The information is used to create a folder structure representing the package.

9. Please specify a module name for your application

   Will be used as a identifier for you module.

10. Please specify a name for the database schema of your application. Use only [A-Z0-9_]*, e.g. EHF_NAME:

    Will be used as the name of the database schema of your module.

    Use only characters supported by the underlying database.

11. Please specify the version of eHF you would like to use:

    This version will be used for the ehf dependencies of your project.

    The initial project structure for the module is generated. The configuration files contain default configuration. The initial model is empty.

12. Import the project into the IDE
    Finally, you can import the project into your IDE. Maven provides plug-ins for different IDEs. For eclipse, e.g., run

    ```
    maven eclipse:eclipse
    ```

    . This will create the Eclipse specific `.classpath` and `.project` configuration files. Open Eclipse and import the project using the **New Project Wizard**.

    The module can be developed using an eclipse-based IDE.

2.2 Create your Domain Model

Create the domain model for your module.

In this step you will create a domain model for your module. The domain model shall contain a custom service class of stereotype ehf-service. This Class will represent your custom web service.

1. Create the domain model
   Use an EMF capable UML tool, like MagicDraw or Topcased, to create an initial model of your module.

   The standard module template provides you with a MagicDraw model `src/main/model/model.xml` and an EMF UML v2.0 file named `src/main/model/model.uml`.

   The initial UML model of your module is available.

2.3 Custom Code Extensions

The generator has generated an extensible infrastructure that can be customized.

1. Extend module with custom code

   You can extend the signature of your interfaces, the implementation of this interfaces and the data models with custom code.

   Moreover all configuration can be manipulated or extended.

   In order to use the full features of the generated eHF architecture consult the eHF Reference Documentation.

   The module contain generated code with custom extensions.

2. Build the module

Execute the command

```
maven dev:build
```

from the root directory of your project.

This triggers a full build of the module. The custom and generated code and configuration compiled and packaged in the module artifacts.

The changes are available in the module artifacts in the local maven repository

2.4 Create Tests

Create unit tests to test your module's behavior

Test your implementation.

1. Create unit tests for your module
   Create a comprehensive suite of unit tests and place them in the directory `src/test/java`. They will be automatically executed during each build.

   A suite of unit tests is avilable ensuring the module functions as expected.

3 References

E

ICW eHealth Framework Reference Documentation
BAS Technology and eHF Committers, InterComponentWare AG (2008-2010)
http://idn.icw-global.com/downloads.html