Helianto Reference Guide

Maurício Fernandes de Castro

v. 0.2.6

What is Helianto

The Helianto Project was born after a discussion on how to leverage the competitive advantage of group of small organizations having different competences in the software business. The key value behind the Helianto mission development was to promote collaboration in disciplines where all members might have common requirements.

The name "Helianto" derives from the Latin word for the sunflower plant. The idea showed up from an original sketch, where a couple of functional modules were drawn in a central corolla, surrounded by as many application "petals" as desired. The name still remains associated to the sunflower geometry, although the project focuses only the corolla.

Helianto is not meant to be a product to be used out of the box. Actually, it requires customization and some expertise in object oriented programming to become a productivity tool. It strongly relies on Spring framework to achieve the majority of its goals. Those who are familiar with Spring's concepts like dependency injection and aspect orientation will be more comfortable here. And those who are not, will find at Helianto a fast and convenient learning path.

Helianto is distributed under the "Apache License v. 2.0".

Currently, Helianto has a work in progress status. This document is currently in the process of being written, so not all topics are covered. Contributions are welcome.

How Helianto may help

Helianto aims to capture common project issues and combine it with sound project practices. Many designers need to model entities like customers or suppliers, payables and receivables, and so forth. Helianto provides extensible domain and service classes for them.

Many designers have a growing interest on ORM. Helianto provides basic configuration for Spring and Hibernate or JPA to achieve this, as well as persistence layer for the supplied domain classes.

Many designers would refactor the code to reduce coupling. Helianto enforces the use of dependency injection. Some have realized EJB is quite complex and will find here how to keep declarative transaction management simple. Some would like to

benefit from a non invasive security framework. Helianto provides configuration and a template for a secure web application using Spring Security.

Helianto strongly relies on Spring to achieve all of this. It also embeds a couple of design decisions that narrow the power of Spring. It is a trade off. In one hand, you have to follow a pre-configured application, but in the other, you may get ready

shortly.

The key value to develop Helianto classes and resources is to keep wide scope and flexibility to ease composition or extension.

In most cases, to create an application backed by Helianto, the developer would:

1. use maven to create a project and resolve dependencies,
2. customize the jdbc connection through the jdbc.properties file (tables in the datastore are automatically created),
3. extend the presentation layer to invoke the service layer methods available through Helianto packages, and
4. test and deploy the application to a servlet container, like Tomcat.

If the application requirements become more specific, the developer would also have to create or extend domain classes, create or extend service facades and wire them to the dependency injection container registry. Throughout this document, many of the aspects of such customization will be described in deeper detail.

Mission

The mission is important for developers and users to keep expectations and development efforts within the same direction.

The Helianto Project Mission

Continuously develop a server-side application base framework that:

1. enforces separation of concerns,
2. is as decoupled as possible from the application and presentation layer, allowing the desired customization to expose service and persistence layers that fits best its purpose,
3. provides an extensible and flexible service layer (SPI and API) to accommodate both simple and complex project requirements,
4. concurrently provides a common domain model to solve well known business problems and a persistence layer as decoupled as possible from the datastore, and
5. enforces good programming practices like design patterns usage, rich documentation and extensive testing.

Acknowledgments

The Helianto project includes software developed by the Apache Software Foundation (http://www.apache.org), Spring Source (http://www.springsource.com) and Hibernate (http://www.hibernate.org), among many others.

Thanks to Atlassian for providing the excellent JIRA issue tracking as an Open Source license.

It is virtually impossible to say thank you to all the people who contributed in some way to the current state of this project. Many of them are project leaders of the packages mentioned

above.

Software Requirements

At compile time:

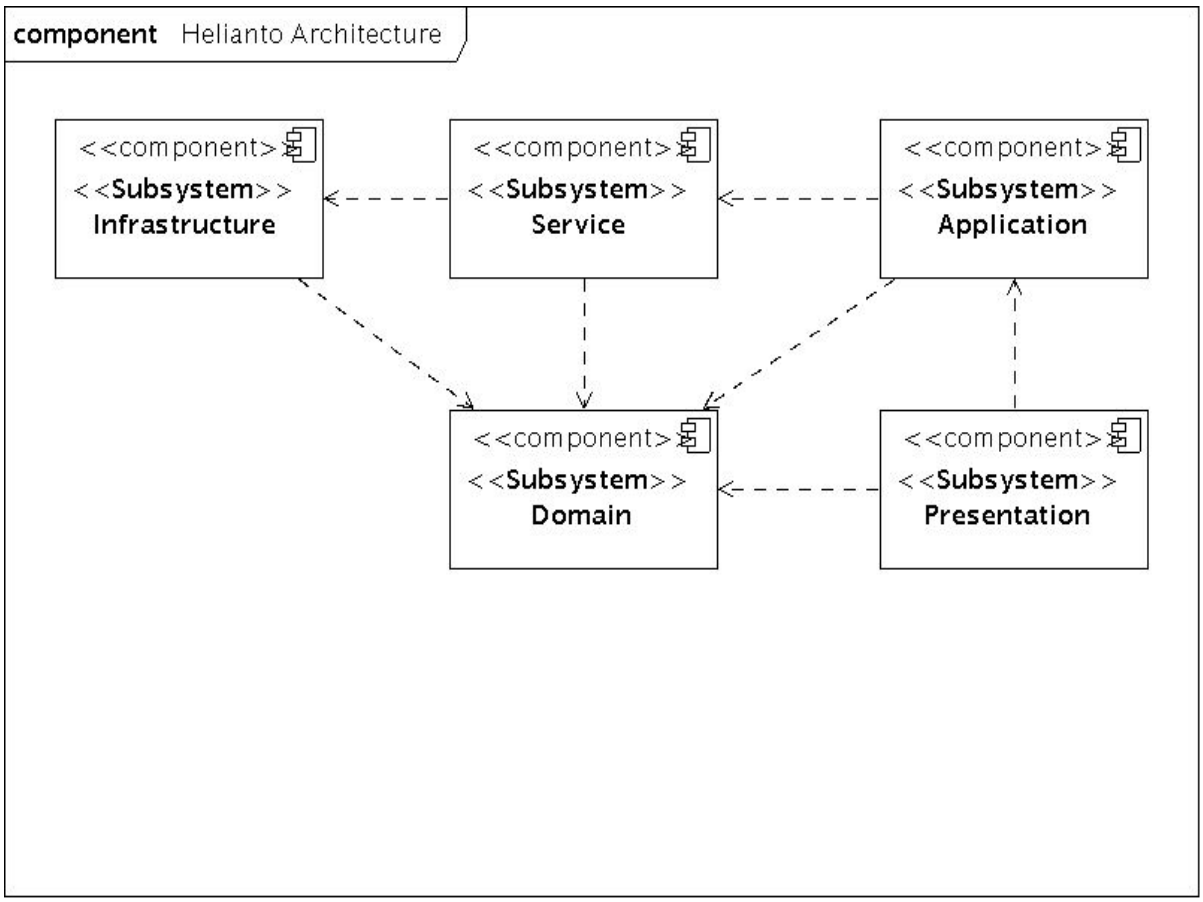
1. The source code includes generics and enumerators, therefore a jdk 1.5 or higher is required.
2. Spring Source STS IDE (or Eclipse platform and the appropriate plugins) as well as Apache Maven (v. 2.x) are recommended, although not required. Of course, there are many dependencies listed in the pom files, and Maven will help to have them ready to use.

At run time:

1. Jre 1.5 or higher is also required,
2. A servlet container, like Tomcat. There is no requirement to an EJB container, although the service layer components can be easily wrapped as EJB's.
3. Any sql database can be used as long as the appropriate properties are overridden in the jdbc.properties file.

General architecture

The high level architecture is shown on the diagram below. Please notice that the dependencies are organized following the principle of separation of concerns to expose only what is going to be consumed on the next level.

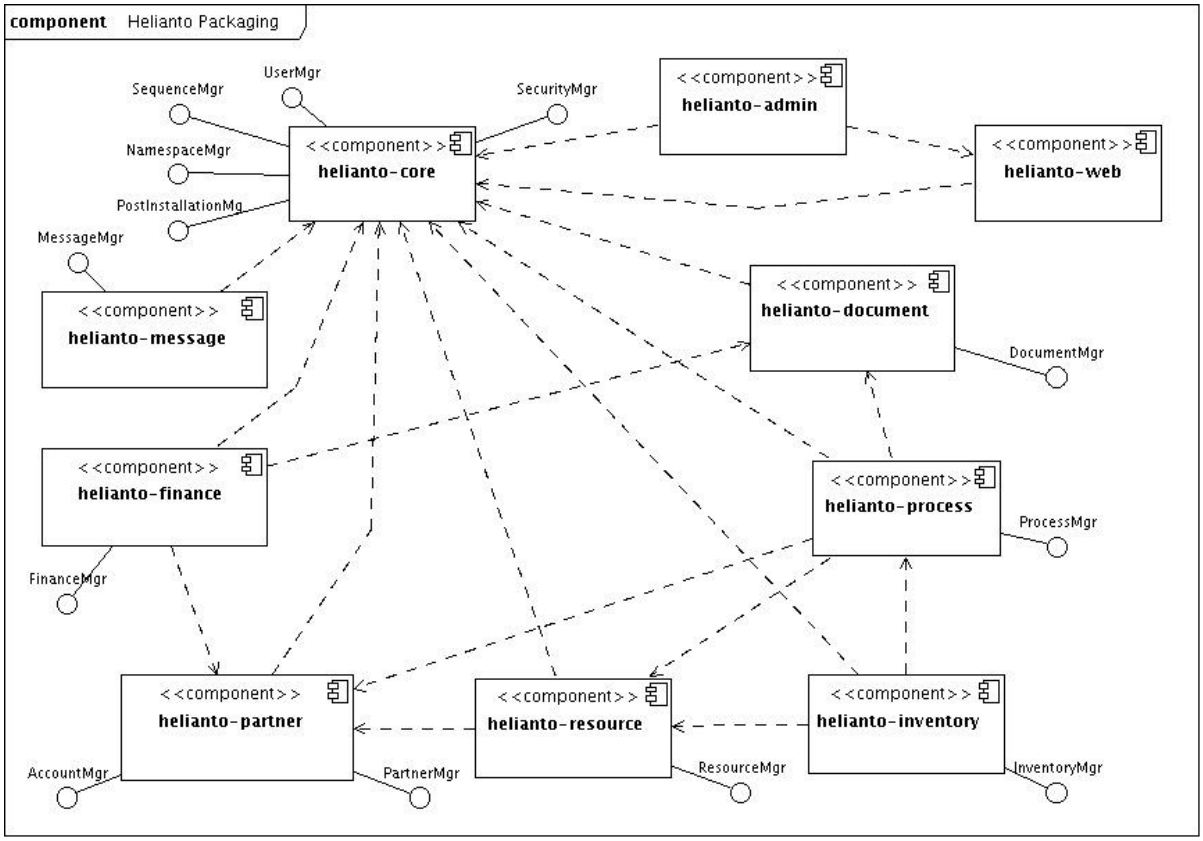


Modules

Here is the existing correlation between modules and the high level architecture:

Module	Domain	Infrastructure	Service	Application	Presentation
helianto-core	Basic isolation classes, security, common classes.	Data sources, ORM wrappers, repositories and factories.	Namespace, sequence, post-installation, security and user management.	Not available.	Not available.
helianto-partner	Customer, supplier and other partner related classes, accounting.	Repositories.	Partner and account management.	Not available.	Not available.
helianto-document	Basic document and control classes	Repositories.	Document management.	Not available.	Not available.
helianto-resource	Resources and groups	Repositories.	Resource management.	Not available.	Not available.
helianto-finance	Cash, payables, receivables, statements, etc.	Repositories.	Finance management.	Not available.	Not available.
helianto-process	Processes, Parts, BOM, etc.	Repositories.	Process management.	Not available.	Not available.
helianto-inventory	Stocks, transactions, etc.	Repositories.	Inventory management.	Not available.	Not available.
helianto-message	Notification classes	Communications infrastructure (e-mail, RSS)	Message management.	Not available.	Not available.
helianto-web	Not available.	Not available.	Not available.	Webflow templates	Freemarker templates
helianto-admin	Not available.	Not available.	Not available.	Basic application navigation.	Admin application.

The next diagram shows module dependencies.



How to create your own project based on Helianto

The easiest way to take advantage of Helianto capabilities is to create a new Maven project in Eclipse (Spring STS comes with all you will need for this) and setup the dependencies in pom.xml like this:

```
<dependency>
        <groupId>org.helianto</groupId>
        <artifactId>helianto-core</artifactId>
        <version>...</version>
</dependency>
<dependency>
        <groupId>org.helianto</groupId>
        <artifactId>helianto-partner</artifactId>
        <version>...</version>
</dependency>
<dependency>
        <groupId>org.helianto</groupId>
        <artifactId>helianto-...</artifactId>
        <version>...</version>
</dependency>
```

Don't forget to replace the ellipsis (...) with the appropriate versions and other module names you may need. If you don't want to install the package manually in you local repository, add also this to your pom.xml:

```
        <repositories>
            <repository>
                <id>org.helianto.sourceforge.repository.release</id>
                <name>Helianto Releases</name>
                <url>http://helianto.svn.sourceforge.net/viewvc/helianto/m
aven/</url>
            </repository>
        </repositories>
```

Please note that snapshots are not uploaded to this repository.

The following chapters will describe in deeper detail the features available in every module.