WhichContainer

Fortress, Phoenix, Merlin, Oh My! Which Container Should I Use?

A common question amongst new Avalon users is "Which container should I use?" or "What's the difference between Fortress, Phoenix, and Merlin?" This section of the wiki has been set aside to help answer these questions. For a more verbose history of where the containers came from, see the ContainerStory.

Short Introduction to Each Container

- **Phoenix**'s strong point are server-like applications. That is, if you want to build a web server, and FTP server, a new EJB container, a chat server, and so on, you may want to try Phoenix. It's stable and there are plenty of existing Phoenix blocks (applications) you can use as starting points. That said, you can also use Phoenix in standalone swing applications, but that's often easier to do with Fortress.
- **Fortress** is a light-weight fully-compliant Avalon 4 container. It's often called 'embeddable' partly because it doesn't come with its own bootstrapper. That is, you'll need to write your own 'main' class to start the container. The advantage is that you can easily start it up in just about any environment – servlets, swing apps, command line apps, etc. It doesn't have all the bells and whistles of Phoenix or Merlin, but it's solid and easy to use. Fortress also replaces the older Excalibur Component Manager, so if you have legacy code using ECM, check out Fortress.
- **Merlin** is the latest and greatest and where a lot of the current action is. Version 3.0 can be downloaded from the standard Avalon download section and there are release candidates available of newer versions which have a number of advanced features you might find hard to live without. Certain Avalon developers will tell you that you shouldn't bother with anything else 😉 , but there's plenty of documentation and several small tutorials available for you to make your own decision.

please see the AvalonContainerFAQ for more information

Meta-Info and Meta-Data

Container Meta-Info Matrix (standard are critical)

Container	Meta-Info Strategy
Phoenix	Container specific <blockinfo> descriptors. Contains information about service publication and service dependencies.
Merlin	Uses the standard Avalon Meta package. Avalon Meta descriptors expose information about service publication, runtime dependencies, context dependencies, deployment dependencies, deployment extension support, and a framework for general attribute association.
ECM	Uses marker interfaces to derive meta-info.
Fortress	Uses a combination of marker interfaces and a service to component mapping table.

Container Meta-Data Matrix (solutions reflect container features)

Container	Meta-Data Strategy
Phoenix	Maintains a list of components under a config file which are assembled based on a static mapping declared under an assembly file. System configuration is archived through a facilities file.
Merlin	Meta data for the kernel definition is contained under a kernel configuration (kernel.xml) which defines internal facilities defined under a system block. Component meta-data used for internal facilities and application components are described in a block directive. A block directives may contain component and container declarations, and include other from local or remote locations. Block directives may be configured to simulate components.
ECM	Uses roles files to map names to configurations.
Fortress	Uses roles files to map names to configurations and provides support for container level configuration.