IoC - Inversion of Control

Inversion of Control (IoC) is the system used to structure trees of Fluid components, to guide their instantiation and resolve references between them. Guides to standard treatments of IoC can be found at Wikpedia and in an important article by Martin Fowler, but the system implemented in Fluid Infusion goes further in power and capability in a number of areas.

The primary goals of our IoC system are to facilitate work by an unbounded community of collaborators on a shared set of software artefacts, and to assist them to extend, reuse and reconfigure the same implementations without running into painful conflicts requiring the use of inefficient reuse techniques such as "cut, paste and modify" or "monkey-patching". Our system for Infusion IoC is crucial to our mission to enable universal accessibility of web applications - since without such a system, it is impossible to deliver the transformation and adaptation of application structures required by a wide community of users with differing accessibility needs, without requiring costly redevelopment, community and code forking, and other punishingly inefficient reuse methods.

Infusion IoC can be more powerful than traditional IoC techniques because of Infusion's universal use of Transparent State, and the overall oversight of the framework in taking complete responsibility for construction of an entire tree of components and subcomponents. Infusion IoC's responsibility does not end at the point of instantiation, but it carries on taking responsibility for resolving references and lifecycle activities until the point of destruction of the component tree. Infusion IoC absorbs a majority of "Design Patterns" into the structure of the framework (most prominently, those relating to creation, structure and sequencing), so that the user's code is not polluted with them. The body of user code is reduced to two main constituents - i) a collection of JSON configuration (typically issued to the framework call fluid.defaults) describing the responsibilities and geometry of a set of components, and ii) a set of free (and ideally pure) functions assembled in appropriate global namespaces. In this form, every element written by the users and integrators is maximally reusable.

IoC is a pattern that naturally results from the proper organisation of dependencies in a body of code. It typically results in delegation of power over instantiation and wiring of dependencies to a framework, rather than writing manual code to sequentially achieve the task in a particular situation. This page provides resources to help you understand the implementation of the Infusion IoC framework and how to use it.

Topics

- Why Inversion of Control
- How to Use Infusion IoC
- IoC References
- Subcomponent Declaration
- Controlling The Timing of Subcomponent Creation
- Dynamic Components
- Contexts
- Expansion of Component Options
- IoCSS: Downward-Matching CSS-Like Context Selectors For Options Forwarding
- Invokers
- Pseudoarguments
- Event injection and boiling