Building an IoC Container

Kevin Jones

@kevinrjones

Overview

- Inversion of Control
  - Using reflection and class loading to build a (simple) container

Creating Objects

```
public class Meeting {
    public Meeting(
       IMeetingService service,
                    String name)
    {}
}
```

Building an Instance

```
public <T> T resolve(Class<T> type) throws IoCException {
        // find type in registrations map
        // find biggest ctor
        // resolve all the ctor params
        // apply any ctor args from map by name
        // create type
}
```

Summary

- Reflection and class loading let us write very flexible applications
  - Inversion of Control is one way of providing that flexibilty
  - Many advantages, for example, making code more testable