Inversion of Control, Ruby and Rails

Adrian Kosmaczewski

2005-07-31

Next week I will be in Belgium working with the Thales team in Brussels, building a new software solution (for a customer of the public sector that I cannot disclose here) using the following technologies:

- Spring Framework for .NET
- NHibernate

Personally, this seems like a rather new (Microsoft-less) way of doing a .NET application, and I like this! To understand what Spring is, I dived into Martin Fowler's Inversion of Control (IoC) / Dependency Injection paper… not an easy trip, believe me; but a rewarding one.

The concept of IoC seems daunting at the beginning, but I found it rather simple (after the fifth or sixth time I read the paper, I must admit). The idea behind it, simply put, is the one that distiguishes a function library from a framework: while the first simply provides a set of disconnected black boxes, encapsulating (useful) functionality, the latter embodies the idea of protocols, callback methods and hooks, that provide not only encapsulated functionalities but also behavior and enforcement of best practices.

The basic idea of an IoC container, simply put, is: **don't call me, I'll call you**. Those of us who have done a little VB event-based programming have already found this pattern, without knowing its name; when you create a VB application (VB6, VB.NET, ASP.NET) the event-based paradigm tells you: don't provide your own event loop; just write the code you want to execute upon a button click on that event handler, and we'll call it up for you when the user clicks on the button. This practice, as simple as it seems, has unlocked the graphical development community (again, this had been invented in a NeXT computer, a couple of years before VB appeared on the market… but that's another story).

Lately, this concept has been adapted to larger enterprise applications through the idea of separating the behavior of the application from its architecture, using configuration files specifying the classes to be loaded at runtime. This way, you can architecture your code in the best possible way (programming against interfaces, using factories, the strategy pattern and other useful design patterns); and then, later, specifying externally the classes to be loaded at runtime. This

can be **extremely** useful in large applications (it usually does not make any sense in small ones), and it provides the ability to change the behavior of the system without having to touch the source code. Extremely powerful, indeed.

This is the description of the Spring.NET Framework, taken from the website:

> Spring.NET is a port of the Java based Spring Framework. Spring for Java contains a lot of functionality and features, many more than Spring.NET currently offers. The initial release of Spring.NET contains a full featured Inversion of Control container. Subsequent releases will contain support for Aspect Oriented Programming (AOP), ASP.NET, Remoting, and data access. This introduction discusses each of the existing libraries in turn.

And this is the description of NHibernate, taken from the home page:

> NHibernate is a .NET based object persistence library for relational databases. NHibernate is a port of the excellent Java Hibernate relational persistence tool.
>
> (If you are courious about what is Hibernate: Hibernate is a powerful, ultra-high performance object/relational persistence and query service for Java. Hibernate lets you develop persistent classes following common Java idiom - including association, inheritance, polymorphism, composition and the Java collections framework. The Hibernate Query Language, designed as a "minimal" object-oriented extension to SQL, provides an elegant bridge between the object and relational worlds. Hibernate also allows you to express queries using native SQL or Java-based Criteria and Example queries.)

The concept of IoC is also at the heart of one of the most surprising, hiped, simple and useful tools that have appeared in the developer scene this year: Ruby on Rails. Based on the Ruby language, which I have already covered in another article (in Spanish), this little framework encapsulates a web server, a persistence engine and an MVC application server, in a small runtime not bigger than 10 MB. **IMPRESSIVE**.

I have gathered here some useful links about the Ruby language and Ruby on Rails, I hope that you find them useful:

Introduction

Ruby is a programming language with the following characteristics:

- Scripting language
- Full OOP (like Smalltalk)
- Dynamic binding (like Objective-C and JavaScript)
- Syntax based on Perl, Eiffel and Ada
- Exception handling
- Garbage collector

- Multithreading
- Large standard library
- Reflection capabilities
- Available for many systems:
  - Windows (95 to 2003)
  - MacOS X
  - OS/2
  - MS-DOS
  - Linux
  - BeOS
  - Many Unix flavors

The creator of the language is Yukihiro Matsumoto, in 1995. The latest stable version is 1.8.2.

Links

- Ruby - The official website
- Ruby in Wikipedia
- RubyForge
- Ruby Central
- Free book
- (Another) Free book

Ruby on Rails

Ruby on Rails (RoR) is both an MVC-based runtime web framework, as well as a set of helper scripts. RoR helps developers build websites without having to use configuration files or an instance of Apache or IIS in their machine. RoR enforces the separation of code following the MVC (Model-View-Controller) design pattern.

Features

- Caching
  - per page
  - per action
  - per fragment
- Data validation
- Database transactions
- Unit testing
- API for creating new generators
- API for security
- AJAX support

Related projects

- For Java: Trails
- For .NET: MonoRail
- By Microsoft: Atlas Project

Sample RoR applications

Rails Day is a competition which gives teams of developers 24 hours to build the best web app that they can using Ruby on Rails. 191 programmers split up into 121 groups competed during the competition contributing more than 18,000 lines of code and 3,335 subversion checkins. These are the winners:

- http://124.railsday.rufy.com/record/list_items
- http://75.railsday.rufy.com/
- http://65.railsday.rufy.com/

Websites using RoR

- http://www.tadalist.com
- http://www.basecamphq.com
- http://www.43things.com
- http://www.snowdevil.ca (Under redesign)
- http://www.cdbaby.com

Links

- http://www.rubyonrails.org/
- http://www.planetrubyonrails.org/
- http://en.wikipedia.org/wiki/Ruby_on_Rails
- Lead developer's weblog
- Tutorial
- Performance compared to J2EE
- Growing up an open source ecosystem

Demo (July 12th, 2005)

On July 12th, 2005, I did a live demo of RoR for Thales Geneva. The demo was based on the contents of the following articles:

- http://www.onlamp.com/pub/a/onlamp/2005/01/20/rails.html
- http://www.onlamp.com/pub/a/onlamp/2005/03/03/rails.html
- http://www.onlamp.com/pub/a/onlamp/2005/06/09/rails_ajax.html