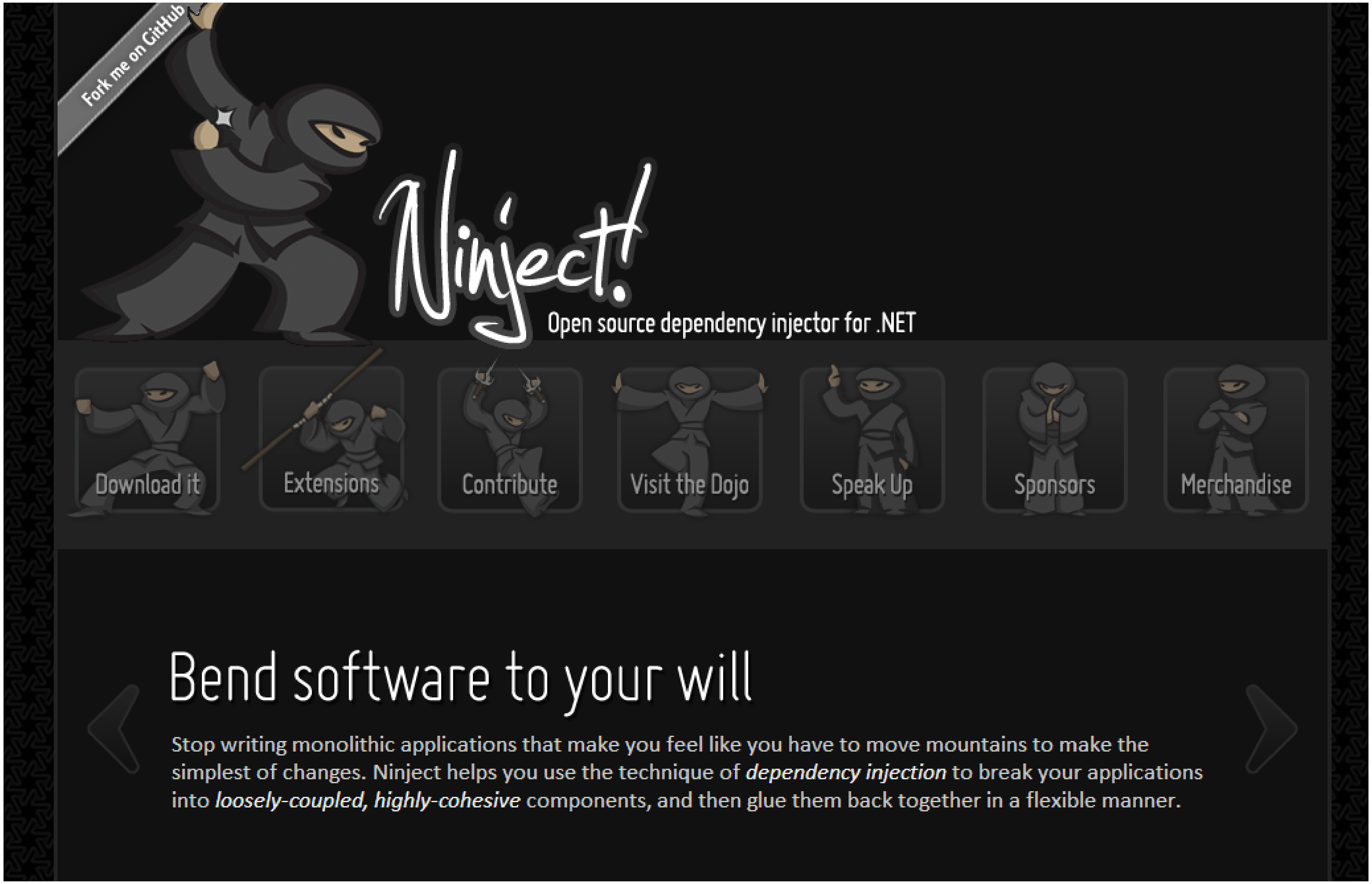
Practical IoC With MVC 4

Using Other IoC Containers

John Sonmez
http://simpleprogrammer.com
@jsonmez

pluralsight
hardcore developer training

Ninject



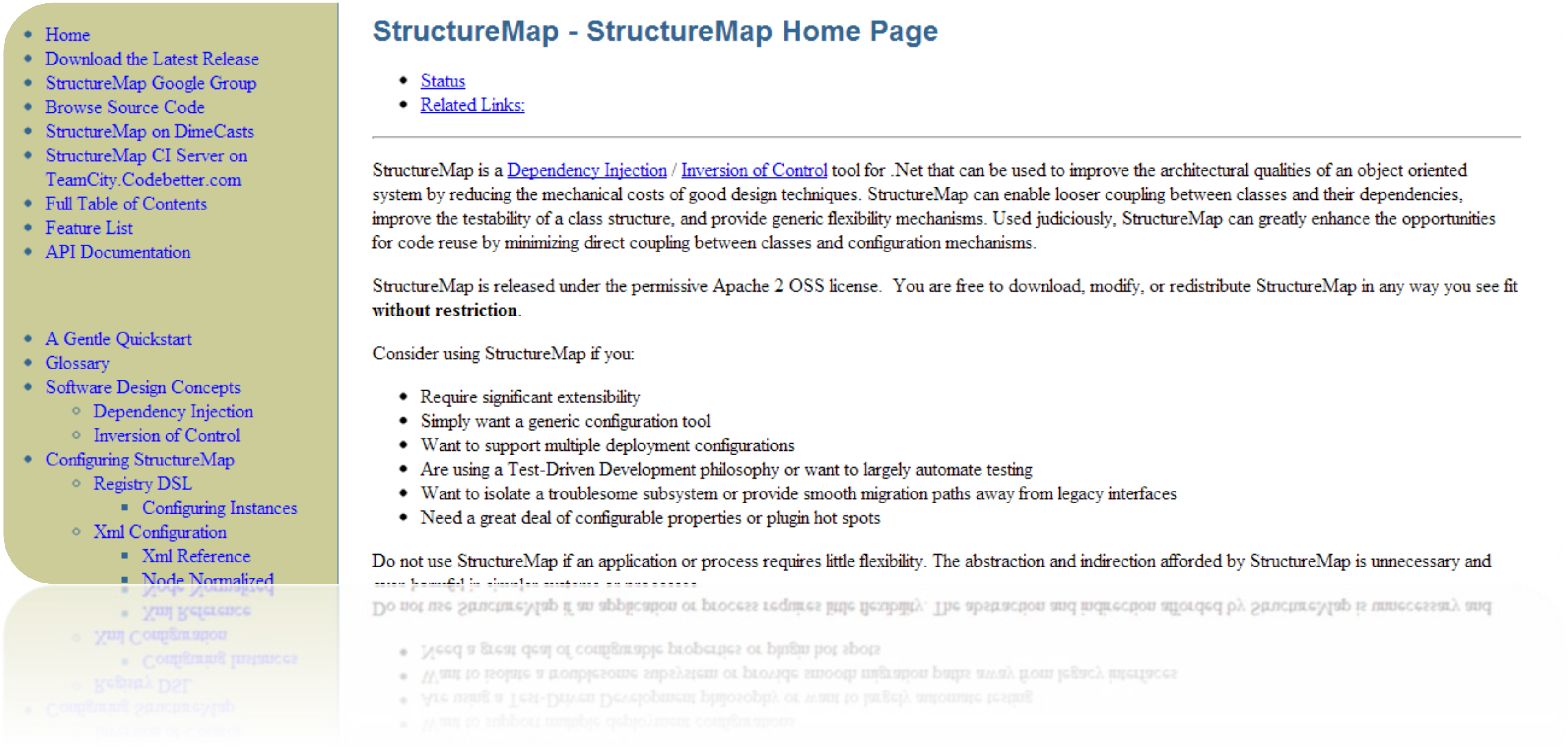
Open source dependency injector for .NET

Bend software to your will

Stop writing monolithic applications that make you feel like you have to move mountains to make the simplest of changes. Ninject helps you use the technique of *dependency injection* to break your applications into *loosely-coupled, highly-cohesive* components, and then glue them back together in a flexible manner.

http://www.ninject.org/

StructureMap



- Home
- Download the Latest Release
- StructureMap Google Group
- Browse Source Code
- StructureMap on DimeCasts
- StructureMap CI Server on TeamCity.Codebetter.com
- Full Table of Contents
- Feature List
- API Documentation

- A Gentle Quickstart
- Glossary
- Software Design Concepts
  - Dependency Injection
  - Inversion of Control
- Configuring StructureMap
  - Registry DSL
    - Configuring Instances
  - Xml Configuration
    - Xml Reference
    - Node Normalized

StructureMap - StructureMap Home Page

- Status
- Related Links:

StructureMap is a Dependency Injection / Inversion of Control tool for .Net that can be used to improve the architectural qualities of an object oriented system by reducing the mechanical costs of good design techniques. StructureMap can enable looser coupling between classes and their dependencies, improve the testability of a class structure, and provide generic flexibility mechanisms. Used judiciously, StructureMap can greatly enhance the opportunities for code reuse by minimizing direct coupling between classes and configuration mechanisms.

StructureMap is released under the permissive Apache 2 OSS license. You are free to download, modify, or redistribute StructureMap in any way you see fit **without restriction**.

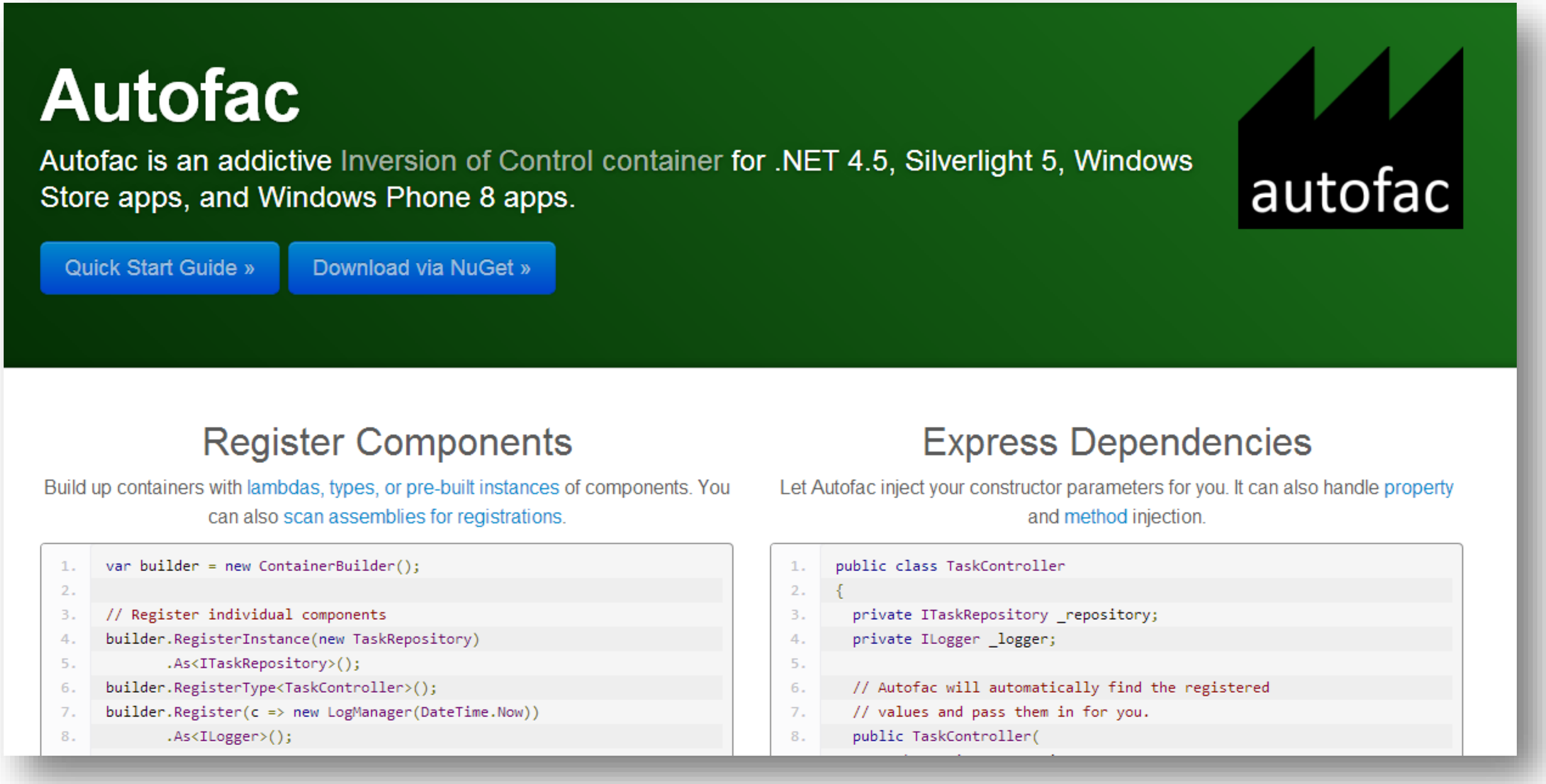
Consider using StructureMap if you:

- Require significant extensibility
- Simply want a generic configuration tool
- Want to support multiple deployment configurations
- Are using a Test-Driven Development philosophy or want to largely automate testing
- Want to isolate a troublesome subsystem or provide smooth migration paths away from legacy interfaces
- Need a great deal of configurable properties or plugin hot spots

Do not use StructureMap if an application or process requires little flexibility. The abstraction and indirection afforded by StructureMap is unnecessary and even harmful in simpler systems or processes.

http://docs.structuremap.net/

Autofac



Autofac

Autofac is an addictive Inversion of Control container for .NET 4.5, Silverlight 5, Windows Store apps, and Windows Phone 8 apps.

Register Components

Build up containers with lambdas, types, or pre-built instances of components. You can also scan assemblies for registrations.

```
var builder = new ContainerBuilder();

// Register individual components
builder.RegisterInstance(new TaskRepository)
       .As<ITaskRepository>();
builder.RegisterType<TaskController>();
builder.Register(c => new LogManager(DateTime.Now))
       .As<ILogger>();
```

Express Dependencies

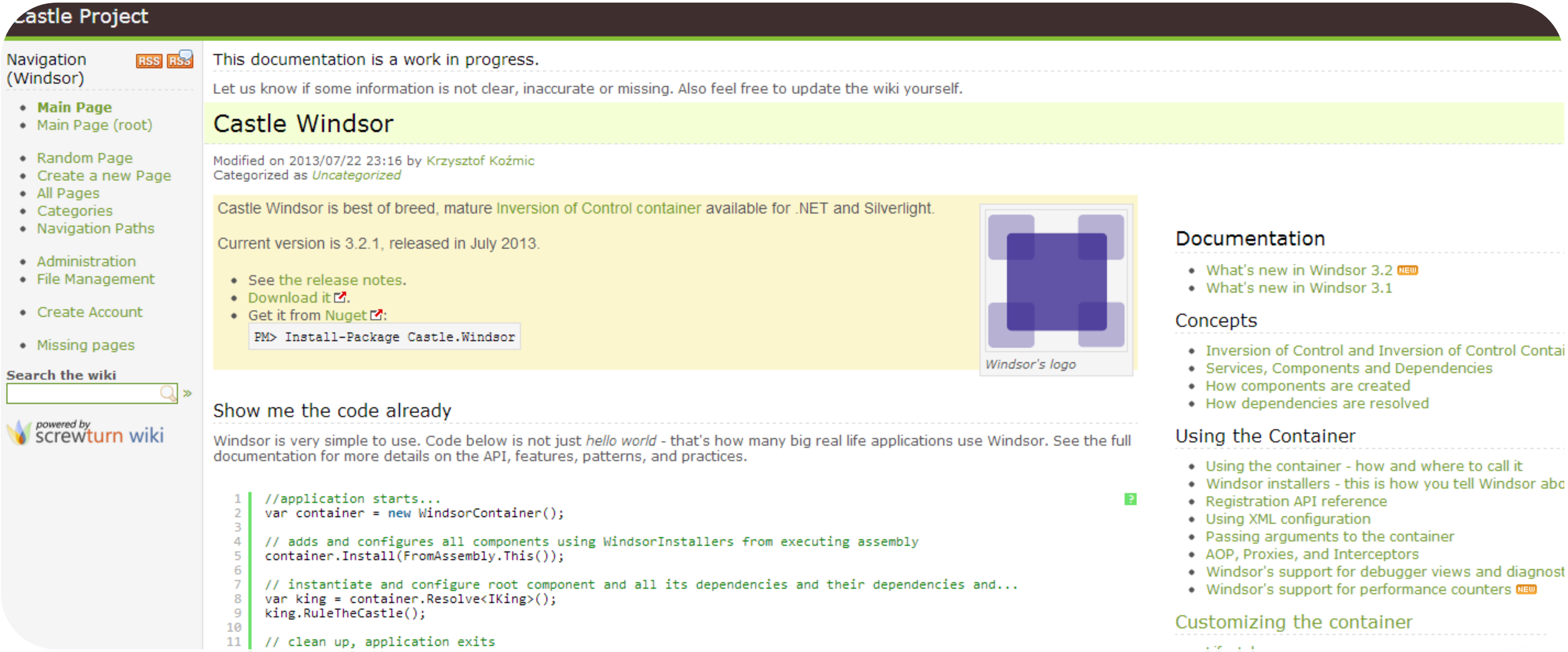
Let Autofac inject your constructor parameters for you. It can also handle property and method injection.

```
public class TaskController
{
  private ITaskRepository _repository;
  private ILogger _logger;

  // Autofac will automatically find the registered
  // values and pass them in for you.
  public TaskController(
```

http://autofac.org/

Castle Windsor



Castle Project

Navigation (Windsor)

- **Main Page**
- Main Page (root)

- Random Page
- Create a new Page
- All Pages
- Categories
- Navigation Paths

- Administration
- File Management

- Create Account

- Missing pages

Search the wiki

powered by screwturn wiki

This documentation is a work in progress.

Let us know if some information is not clear, inaccurate or missing. Also feel free to update the wiki yourself.

Castle Windsor

Modified on 2013/07/22 23:16 by Krzysztof Koźmic
Categorized as *Uncategorized*

Castle Windsor is best of breed, mature Inversion of Control container available for .NET and Silverlight.

Current version is 3.2.1, released in July 2013.

- See the release notes.
- Download it.
- Get it from Nuget:

```
PM> Install-Package Castle.Windsor
```

Windsor's logo

Show me the code already

Windsor is very simple to use. Code below is not just *hello world* - that's how many big real life applications use Windsor. See the full documentation for more details on the API, features, patterns, and practices.

```
//application starts...
var container = new WindsorContainer();

// adds and configures all components using WindsorInstallers from executing assembly
container.Install(FromAssembly.This());

// instantiate and configure root component and all its dependencies and their dependencies and...
var king = container.Resolve<IKing>();
king.RuleTheCastle();

// clean up, application exits
```

Documentation

- What's new in Windsor 3.2 NEW
- What's new in Windsor 3.1

Concepts

- Inversion of Control and Inversion of Control Contai
- Services, Components and Dependencies
- How components are created
- How dependencies are resolved

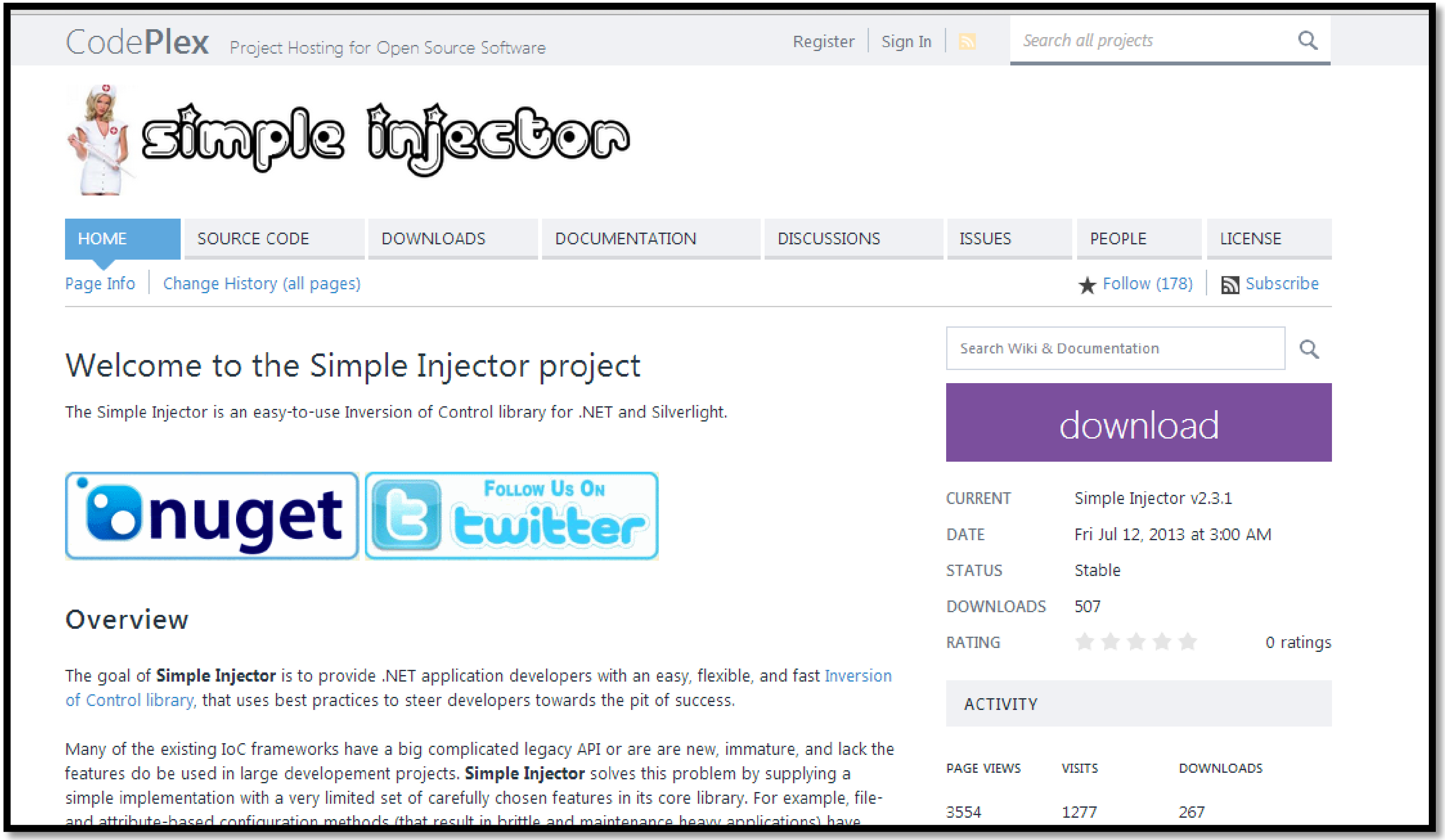
Using the Container

- Using the container - how and where to call it
- Windsor installers - this is how you tell Windsor abo
- Registration API reference
- Using XML configuration
- Passing arguments to the container
- AOP, Proxies, and Interceptors
- Windsor's support for debugger views and diagnost
- Windsor's support for performance counters NEW

Customizing the container

http://docs.castleproject.org/Windsor.MainPage.ashx

Simple Injector



Welcome to the Simple Injector project

The Simple Injector is an easy-to-use Inversion of Control library for .NET and Silverlight.

CURRENT	Simple Injector v2.3.1
DATE	Fri Jul 12, 2013 at 3:00 AM
STATUS	Stable
DOWNLOADS	507
RATING	☆☆☆☆☆ 0 ratings

PAGE VIEWS	VISITS	DOWNLOADS
3554	1277	267

Overview

The goal of **Simple Injector** is to provide .NET application developers with an easy, flexible, and fast Inversion of Control library, that uses best practices to steer developers towards the pit of success.

Many of the existing IoC frameworks have a big complicated legacy API or are are new, immature, and lack the features do be used in large developement projects. **Simple Injector** solves this problem by supplying a simple implementation with a very limited set of carefully chosen features in its core library. For example, file- and attribute-based configuration methods (that result in brittle and maintenance heavy applications) have

http://simpleinjector.codeplex.com/

The End

“These dependencies will keep you healthy and decoupled”