Jeffrey Palermo 8:19 am on August 19, 2013

Tags: onion architecture (4)

Onion Architecture: Part 4 – After Four Years

In 2008, I coined a new pattern name called Onion Architecture. You can read the previous parts here: part 1, part 2, part 3. Over these four years, I've spoken about this pattern at user groups, conferences, and it's even published in one of the chapters of ASP.NET MVC in Action from Manning.

I've been overwhelmed by the traction this pattern name has enjoyed. Folks from all over the country have written about and have talked about the pattern. Some of the ones I've noticed are here (please comment with more – I welcome it).

- Ayende Rahien's opinion on Onion Architecture
- "Cunning" Clinton's view
- Tony Sneed's view and code sample
- StackOverflow questions on Onion Architecture
- Matt Hidinger's Chicago .Net User Group video presentation on OA
- One interesting StackOverflow question.

Back in 2008, I defined four tenets of Onion Architecture:

- *The application is built around an independent object model*
- *Inner layers define interfaces. Outer layers implement interfaces*
- *Direction of coupling is toward the center*
- *All application core code can be compiled and run separate from infrastructure*

Although there has been significant adoption of this pattern, I have received countless questions about how to implement it in various environments. I mostly get asked about how it relates to domain-driven design. First, onion architecture works well with and without DDD patterns. It works well with CQRS, forms over data, and DDD. It is merely an architectural pattern where the core object model is represented in a way that does not accept dependencies on less stable code.

CodeCampServer was an original sample of onion architecture, but it also grew as a sample of how to do ASP.NET MVC in various ways, how to use Portable Areas, and how to use MvcContrib features like input builders. If you are just looking for onion architecture, it has too much going on. I have pushed a much simpler solution that represents onion architecture concepts. I have intentionally not included a UI input form or an IoC container, which most people associate with onion architecture. Onion architecture works just fine without the likes of StructureMap or Castle Windsor. Please check out the code here and let me know if this presents a simple approach – that is the goal.

When there is enough interest, I will continue this series with more parts. CQRS definitely deserves some addressing within this architecture, and so do object models that support task-based UIs.

Get the code here at my BitBucket repository.