BEST PRACTICES ►
SHAREPOINT CONFERENCE

Clarity. Direction. Confidence.

Governing Development in SharePoint

DEV207
Robert L Bogue, MS MOSS MVP

► Robert Bogue, MOSS MVP, MCSE, MCSA: Security, etc.
  - Thor Projects
  - http://www.thorprojects.com
  - Rob.Bogue@ThorProjects.com
  - Latest Book: The SharePoint Shepherd's Guide for End Users

BEST PRACTICES SHAREPOINT CONFERENCE

Agenda

► What can go wrong?
► What is code (and what isn't)?
► Big areas
  - Core Development
  - Checkpoints
  - Deployment

BEST PRACTICES SHAREPOINT CONFERENCE

What can go wrong?

► Security
  - Breaches
  - Difficult to maintain
► Stability
  - Memory Leaks (Out of Memory issues)
  - Endless loops
► Performance
  - Large List
  - Iterating

BEST PRACTICES SHAREPOINT CONFERENCE

What's Code (and what isn't)?

► More than C#/VB.NET
► What about XML?
  - Feature.xml
  - Manifest.xml
  - ONet.xml
  - Schema.xml
► What about Master Pages?
► What about Page Layouts?

BEST PRACTICES SHAREPOINT CONFERENCE

What are the big areas to work on?

► Core development
  - Coding Standards
  - Supported/Unsupported APIs
► Checkpoints
  - Automated analysis
  - Build Servers/Continuous Integration
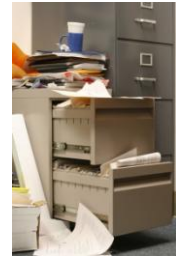► Deployment

BEST PRACTICES SHAREPOINT CONFERENCE

Core Development

Source Control

- ▶ Fundamental of software development – yet often overlooked
- ▶ Verify by requiring builds come from source control, not a developer

IDisposable

- ▶ Every object that supports IDisposable must be disposed every time.
- ▶ Use the 'Using' statement in C# or a try/catch/finally to make sure that .Dispose() is **always** called.

IDisposable - Enumeration

- ▶ SharePoint Enumerators don't automatically dispose objects – put a .Dispose() at the end of your foreach loop.
- ▶ SPSite will dispose all SPWebs it creates but don't rely on it.

Sidebar: StringBuilder (your .NET friend)

- ▶ Memory fragmentation is your enemy (even in 64 bit)
- ▶ In .NET each string is immutable. String concatenation results in a new memory allocation
- ▶ Use StringBuilder

Customized Artifacts

- ▶ Think: SharePoint Designer
- ▶ Customized pages don't upgrade well
- ▶ Customized pages don't distribute well
- ▶ Should you allow customized artifacts?

Site Definition or Site Template

- Site Definitions are **hard**
- Site Templates are **easy**
- Use site definitions only when you must
- If you use a site definition, build it from features

Feature Stapling

- Site Definitions can not be changed once a site is created from it (Supportability)
- Feature stapling allows you to add new functionality to existing site defintions
- Don't allow stapling to GLOBAL or OOB site definitions
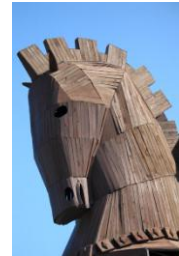
List Definitions and Content Types

- Use Content Types (**easy**) to define fields
- Use List definitions (**hard**) only if you need to pre-create views (and can't use a list or site template)

Event Receivers and Workflows

- Workflows are long running, serializable, and resumable but heavy and not high performance
- Event receivers are not for long running operations, can cancel actions, but are high performance

SPList vs. Database Table

- SPList offers Alerts, searchability, event recievers, workflows, MS Office integration, etc., but isn't for large data sets (> 2K)
- Database tables (not in the SharePoint DB) are easy to understand but don't offer SharePoint features

RenderContents vs. CreateChildControls

- Don't use RenderContents – it's like doing ASP in ASP.NET
- CreateChildControls allows you to modify the output for mobile devices, supports View State, Post Back, events, etc.

Supportability Deal Breakers

- Just **don't** do it
- **Don't** directly access the SharePoint DB (even just reads)
- **Don't** change a site definition after a site has been created

Don't Modify Out of the Box Files

- Don't modify out of the box files (including site definitions)
- If you **must** remember that you have to manually integrate every service pack, update, etc.

Branding

- Use Master Pages to change the fundamental layout
- Use Themes to apply basic colors and graphics
- Make your theme link (import) from _layouts to avoid the customization problem
- Use Page layouts to rearrange content, hide navigation, etc.

Strong Naming

- All Assemblies must be strong named
  - You never know when you're going to have to put them in the GAC
  - It **doesn't** hurt to do it now
  - You don't want to change assembly references later (they're in the DB)

Checkpoints

Automated Checks

- Compiler Warnings
- Visual Studio 2008 Code Analysis/FxCop
- SPDisposeChecker

Build Sever

- All code comes from source control
- Ensures that all configuration dependencies are documented
- Easy to move to continuous integration

Manual Code Reviews

- The most expensive – and necessary – step for shared platforms
- Must be an architect doing the review
  - Look for best practices (i.e. this presentation)
  - Look for infrastructure issues (i.e. not in this presentation)

Load Testing

- At the very least, try to break it (codeplex.com/???)
- Invest in real load testing for high risk projects
- Load testing needs to be done against a representative data sample

User Acceptance Testing

- **Make** the users sign off on the solution – every time
- Ensure that the test matches the requirements, design points, or features of the system

Deployment

SharePoint Solutions (WSP)

- Only allow WSPs for deployment – no manual file copies
- Think MSI for SharePoint

Global Assembly Cache or BIN

- Some things **must** be in the GAC (Event Receivers, Feature Receivers, Workflow DLLs, etc.)
- In BIN require CAS policies (don't turn CAS off)
- In an organization the requirement for BIN is normally not worth it.

Web.Config Changes

- Only if required
- Don't use the API to do them
- No application settings (too hard to manage) – there are better places for application settings

Smoke Tests

- Quick tests that the infrastructure team can use to ensure that the package / solution deployed correctly

Migration Strategies

- Code moves from developer workstations forward through test and into production.
- Content (including configuration) move from production through test and if necessary into developer workstations.

Say What?

- Prioritize
- Evaluate risks
- Educate don't legislate
- Develop strong large scale ASP.NET skills

BEST PRACTICES™
SHAREPOINT CONFERENCE

February 2-4, 2009
San Diego, CA
Clarity. Direction. Confidence.

Post conference DVD with all slide decks

Sponsored by
echoTechnology
Manage SharePoint

Thank you for attending!