www.plonesolutions.com

Top 20 Plone Pitfalls

And Then Some

Stefan H. Holek
stefan@plonesolutions.com

All disclaimers apply.
I am not going to argue over any of these, don't try.

- Do not put off learning Python until [insert favorite excuse here].

Pick up a book RIGHT NOW.

- Python *is* the scripting language.
- Zope and Plone are *written* in Python.
- You can learn Python in a week.
- http://diveintopython.org

2 days if you already know how to program.
I am not talking about metaclasses, decorators, or other advanced concepts.

2

- Do not attempt to outsmart Zope and Plone; you *will* fail.

- Do stuff The Plone Way™.
- If you feel what you are doing smells, stop right there.
- There *is* an elegant solution for everything.

- Do not guess.

3

- plone.org/documentation, zopewiki.org, zopelabs.com, www.neuroinf.de/LabTools, ...
- google.com
- plone-users, plone-setup, ... (gmane)
- #plone
- DocFinderTab, grep, find

There are books even!

4

- Do not think in terms of URLs.
- Corollary: Do not think in terms of websites, even.

- With “traditional” web-frameworks URLs is all you have.
- With Zope, you are building an object system, and the relation of your objects is what counts.
- A URL is just one of the many attributes an object has.
- The fact that your objects publish nicely to the web is simply a side-effect of using Zope.

- Do not build URLs by hand.
- Corollary: Do not assume URLs have any relation whatsoever to the physical path.

Constructing URLs by string concatenation or from the physical path is an effective way to break virtual hosting.

- The path of an object is a function of the ZODB layout, the URL is a function of the current REQUEST.
- An object can have more than one URL at any time.
- The only way to convert back and forth are the REQUEST.physicalPathToURL and REQUEST.physicalPathFromURL APIs.
- brain.getURL already does the right thing.

- Do not develop in release mode.
- Corollary: Do not run a production system in debug mode.

- Debug mode is slower, but automatically refreshes skins, External Methods, e.a.
- You also get better logging.
- Release mode is faster, but you have to restart Zope for changes to take effect.

Zope 2.7 and 2.8 used debug mode by default.

7

- Do not expose ZServer to the Internets.

- Apache
- lighttp
- IIS
- Squid
- Varnish

8

- Do not develop your application in TALES.

Templates section starts here.

- Templates are for presentation, *and for presentation only*.
- Logic goes into Python code (scripts, tools).
- Pay 5 € (or $) into the kitty for every use of "python:"

9

- Do not change strings in templates to rename things.

- Plone has very sophisticated translation facilities.
- Use the .po files, Luke.

plone-en.po

- Do not change settings of filesystem templates (skins) in the ZMI.

The fact that this works at all is probably a design error, even.

- Skins are not persistent, so these settings won't stick.
- Use the .metadata files to make settings.

- Do not impose a relational data model on an object database.

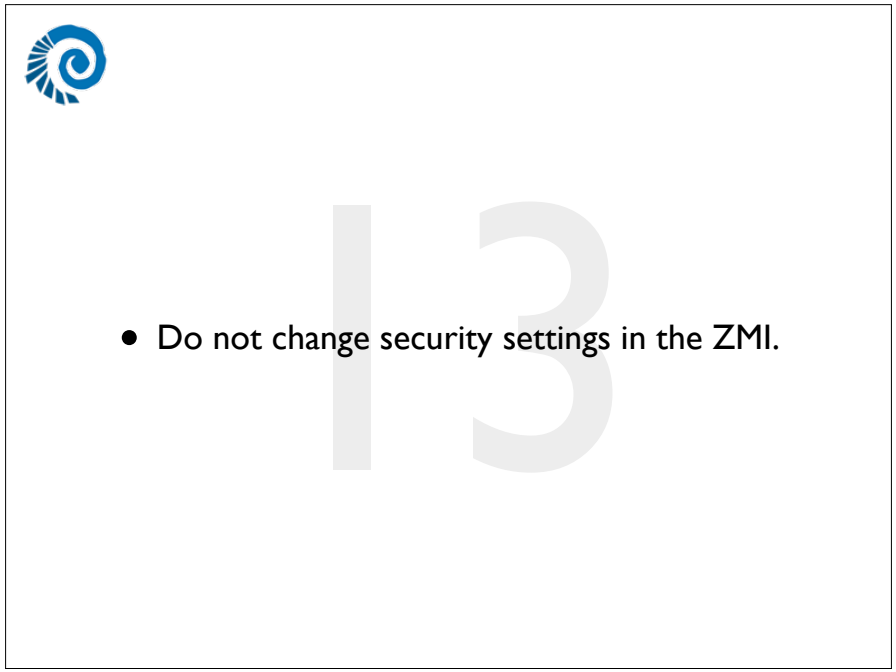
- Do not expect the ZODB to work like MySQL, it won't.
- There is nothing wrong with using an RDBMS with Plone, if your data calls for it.

ZODB section starts here.

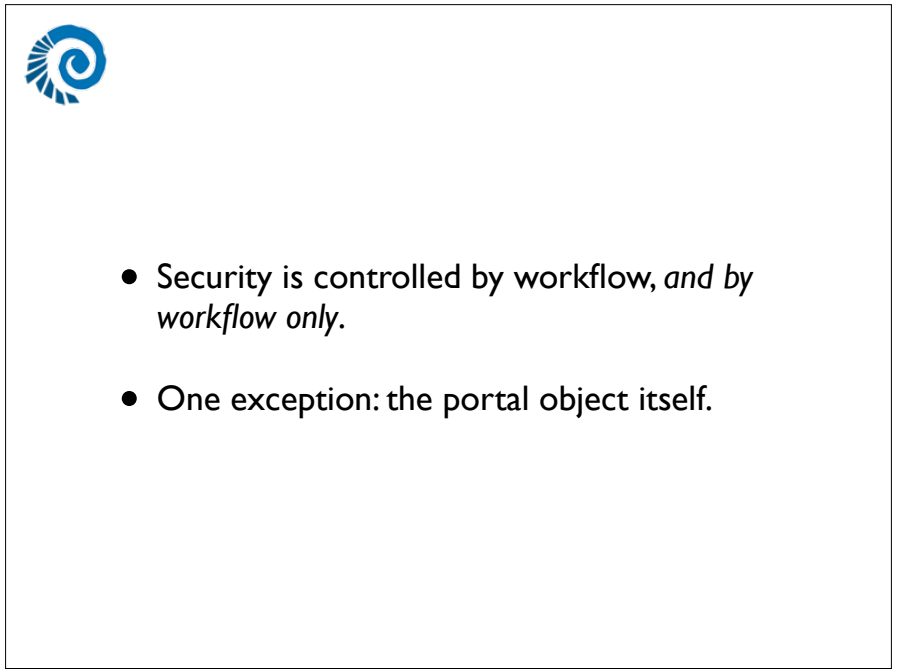
12

- Do not suppress ConflictErrors; avoid bare except: and don't use hasattr (!).

- hasattr swallows *all* exceptions.
- getattr(ob, 'name', marker) is not marker
- CMFPlone.utils.safe_hasattr



Security section starts here.



The Security tab is off-limits! Don't go there, you'll regret it.

- Do not check for roles, only check for permissions.

- That's because objects and methods are protected by permissions, not roles.
- portal_membership.checkPermission.

You need permission to do something.
Roles are responsibilities.

15

- Do not use REQUEST.AUTHENTICATED_USER.

- AUTHENTICATED_USER is unsafe and has been deprecated many winters ago.
- portal_membership.getAuthenticatedUser

16

- Do not use the Authenticated role to model your site's security.

- Authenticated is a system-owned role.
- Add your own custom role(s).

17

- Do not assign users or groups the Owner role globally, ever.

- Owner is *only* useful as a local role.

18

- Do not use proxy roles.

- Proxy roles are similar to SUID scripts in *nix.
- You are poking holes into your site's security; be *very* careful about what your proxied scripts do.

Ask your sysadmin what he thinks about gratuitous use of SUID scripts.

19

- Do not delete user accounts that may own objects in your site.

- Every object has an "owner", typically the user who created it.
- *Not* the same as the Owner local role!
- Zope security intersects the owner's permissions with the authenticated user's permissions.
- When the owner goes away you are in trouble.

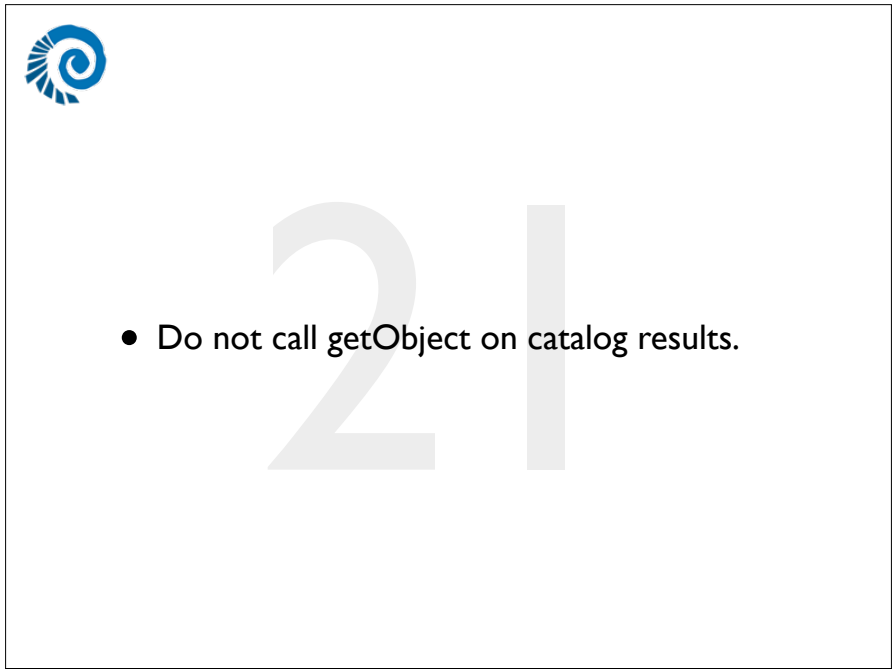
"Executable ownership" introduced to avoid trojan horse attacks. Take ownership of orphaned objects.

20

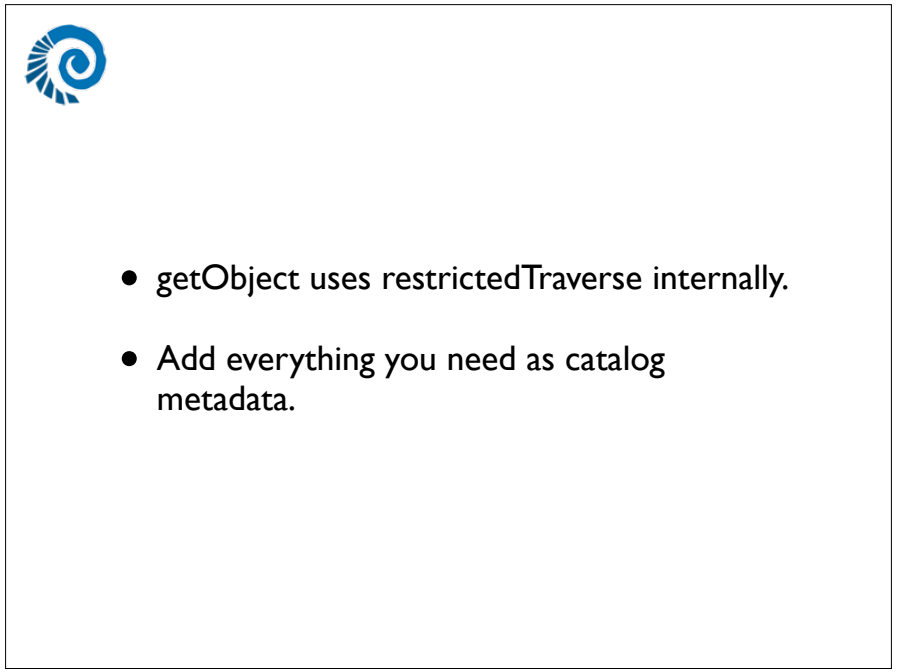
- Do not forget to add security declarations to your methods, the default is public (!).

- That's because Item, which is the base class for most Zope2 objects, allows access to unprotected attributes.
- Be extra careful with tools. Methods may need to be public and perform their own security checks.

We had troubles in Plone recently because of this. See Hotfixes.



Performance section starts here.



The catalog only stores the physical path.

22

- Do not use contentValues nor objectValues.

- contentValues/objectValues "wake up" sub-objects, i.e. load them from disk into memory.

objectIds is OK, incidentally, not however contentIds.

23

- Do not take the truth value of objects, ever.

- “if foo:” is *absolutely* evil, unless foo is of a simple type.
- <tal:condition=”foo”> is even worse, unless foo is of a simple type.

You never know what is happening behind the scenes.
Python tries __nonzero__ then __len__.
condition=”foo” may well do things like render templates.

24

- Do not return objects from scripts, return (lists of) dictionaries.

- Objects cannot be RAM-cached.
- Objects will be security checked.

Python scripts are prime targets for RAM-caching.

25

- Do not compute values at display time.

- Compute values at edit time and store them.

- Do not "touch" objects at display time.

- Viewing an object must not cause a database write.
- Easy to accidentally cause a modification; watch the Undo tab.

27

- Do not optimize code without a profiler.

- Guessing does not work with optimization.
- CallProfiler, PTProfiler, and ZopeProfiler.

- Do not run production sites without a cache in front.

- “Plone should be faster”.
- Plone is fastest when not hit at all.
- mod_cache, Squid, Varnish, ...

Thanks!

Continental Airlines