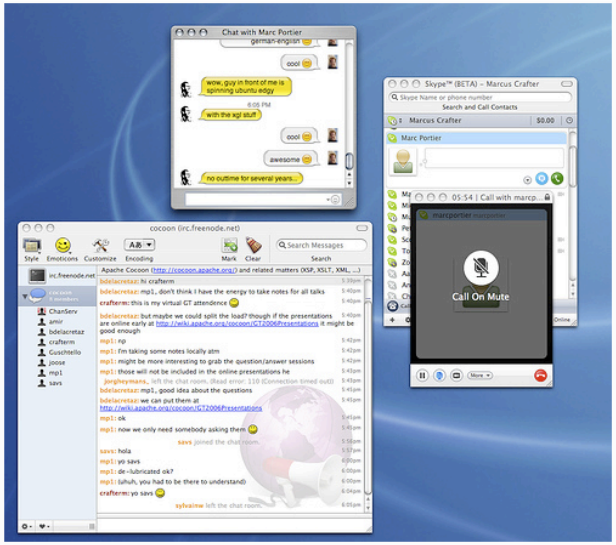
Random notes to test the view
Lorem ipsum

My name's Andrew, it's a pleasure to be here.

It's also a pleasure to see so many familiar faces, friends old and new. Time is short, so I'll get straight into it.

Oh, just before I start ...

GT2006 Virtual Attendence :)



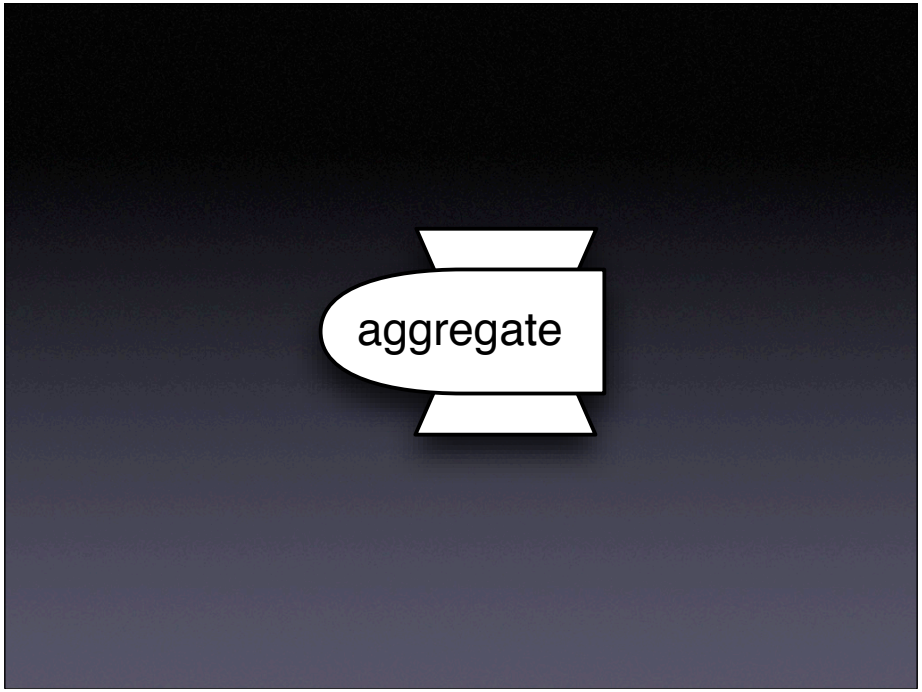
My virtual Cocoon GetTogether attendence this year, the first GetTogether that I'll miss ever - kudos to MPO for letting me hear everything through Skype :)

When I submitted all 6 talks, I never thought for a moment they would all be chosen. I'm not sure the Cocoon community would be too happy about attending the Andrew GetTogether.
Thankfully the people on the mailing lists agreed, and so we've saved some for articles hacked at the hackathon.

So I'm going to drop three of the talks, I hope you don't mind. There's only 30 minutes after all. Or, to put it another way, a one time special offer, I'm giving you 50% off. For free. How nice am I?

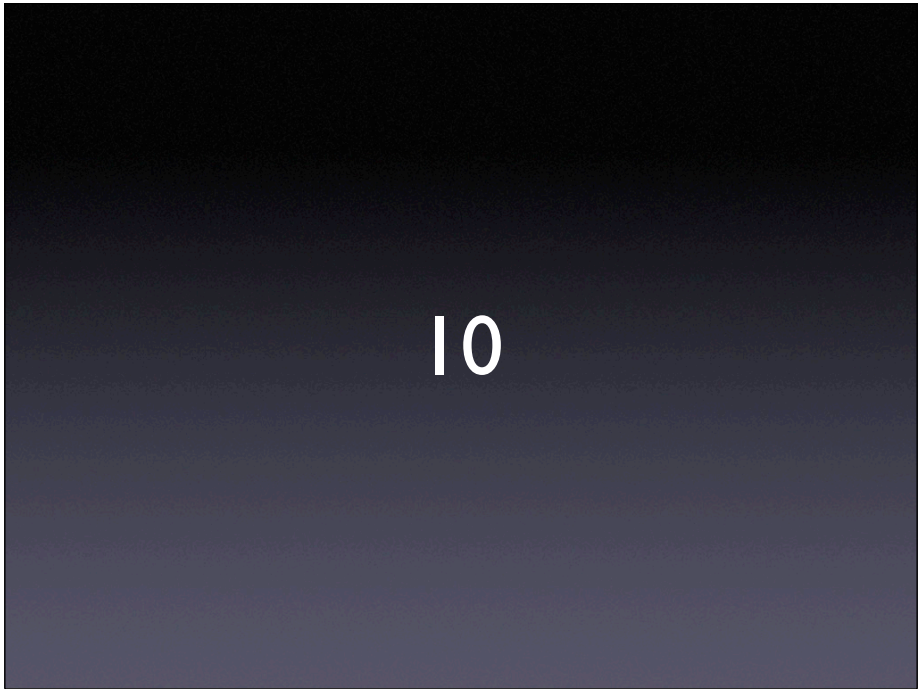
So, without further ado, I proudly present “3 talks on Cocoon”.

The talks I chose were these, which seemed to be the most popular.

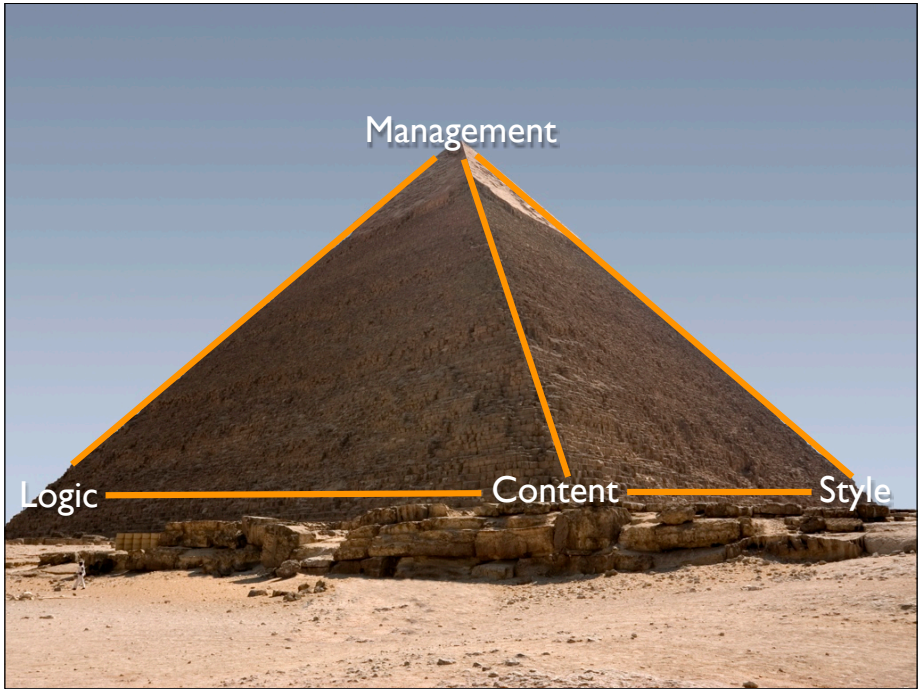


But hang on a minute! This is Cocoon we're talking about here! Why don't we use aggregation, and combine all these talks into one pipeline? Brilliant! Genius!
So, I'm happy to present ....

(rush) 10 things your mother never told you about how to use Cocoon technologies
(BREATH)
illustrated
Time is short, let's get going.
Did I really just use 5 minutes for the introduction? I could have fitted another talk in there! Anyway...


10

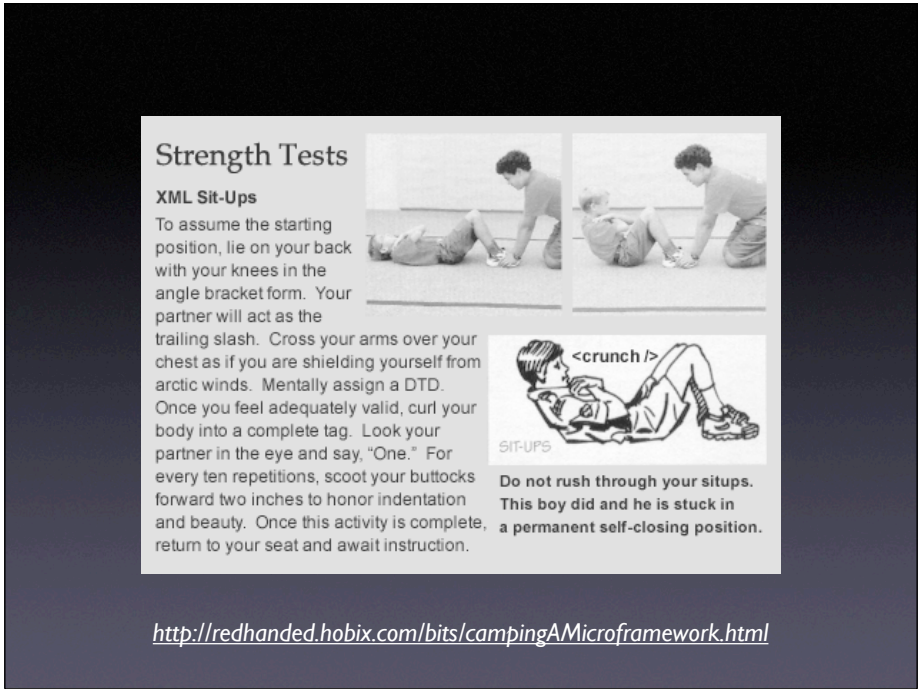
If you’re new to Cocoon, this is where it all begins: Cocoon is built to help you enforce separation of concerns.



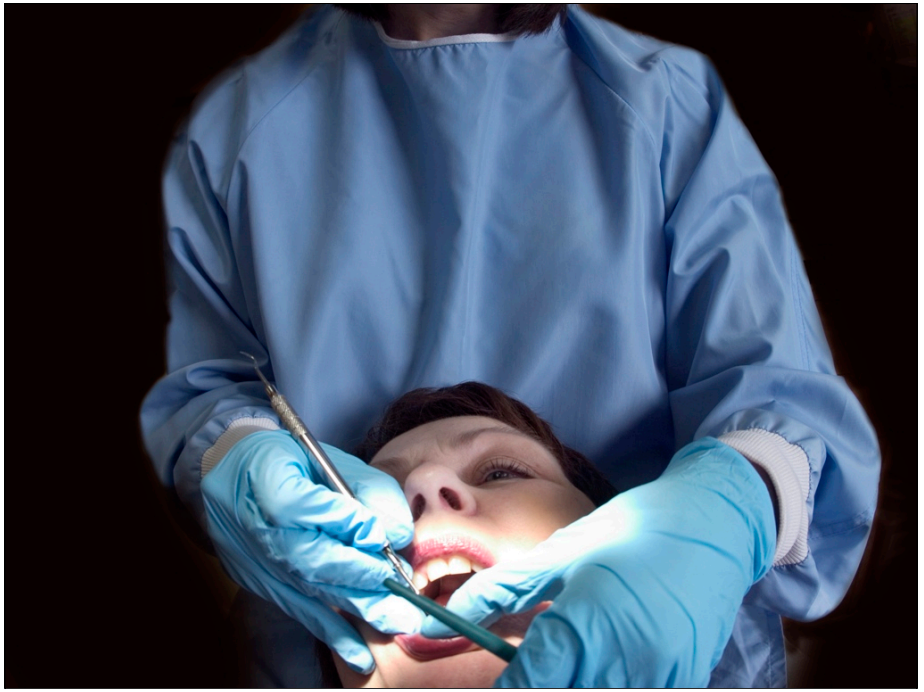
Separation of concerns can be illustrated by the Pyramid of Contracts - 4 major concerns with 5 contracts between them.

After all these years, still nothing keeps these separate like a _good_ cocoon application. Word of warning: it's too easy to mix them, at which point Cocoon's no better than any other badly-used tool.

9



The Ruby on Rails folk talk about XML sit-ups.
You may remember me going on about it last year too.

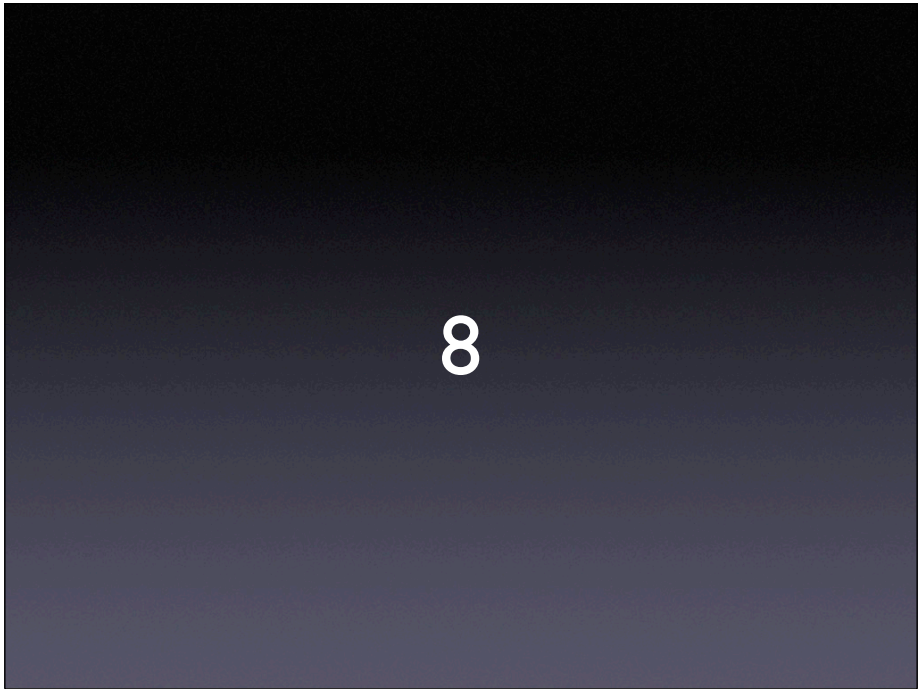


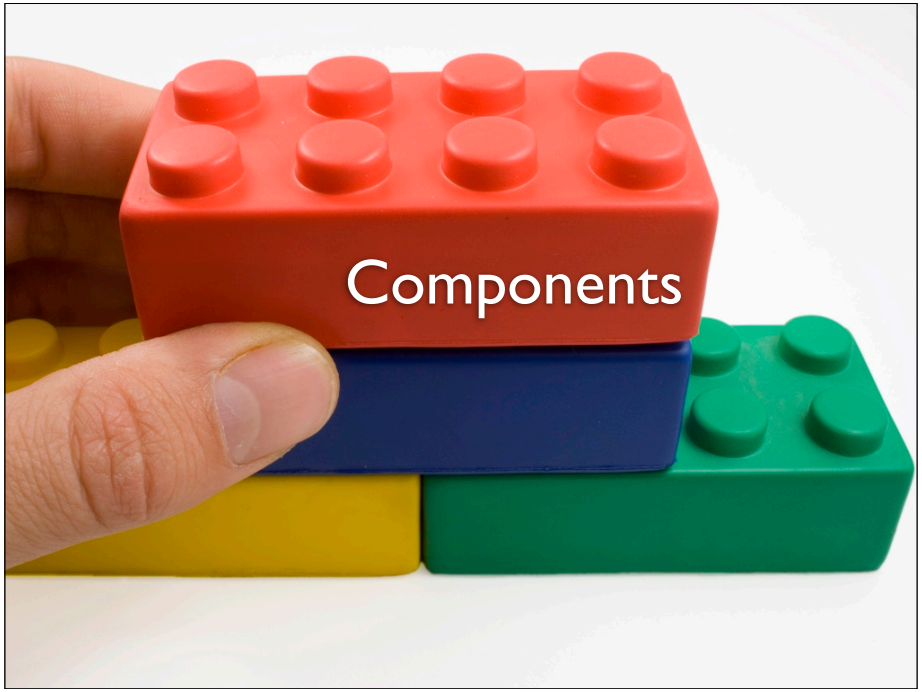
What would you rather do?
Which is better: writing XML in Ruby (or any other scripting language) or just writing the XML?
Call me old-fashioned, but whatever happened to “the right tool for the job”?

Writing XML in anything other than XML reminds me of something ...

When you're working with XML data, pick the right tool for the job.

It's a STRENGTH not a WEAKNESS! Embrace the XML side.

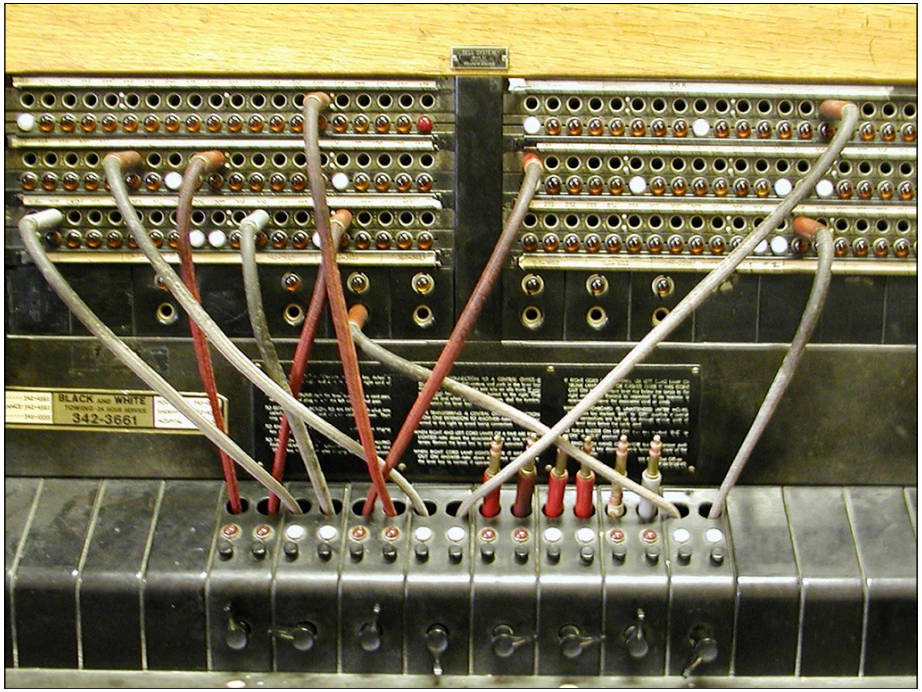

8



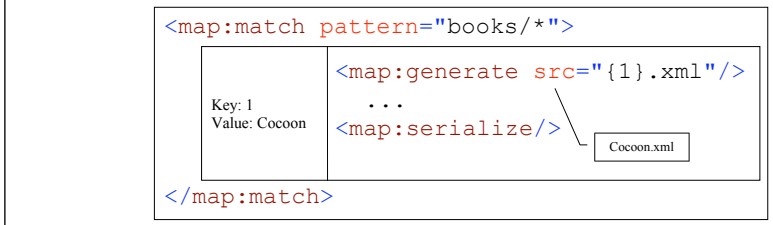
No matter what your application, being able to plug any of the hundreds of pre-written Cocoon components together to build a comprehensive solution is one of THE major advantages of Cocoon. I quote: “I wrote a CRUD application for an LDAP addressbook in one day!”.

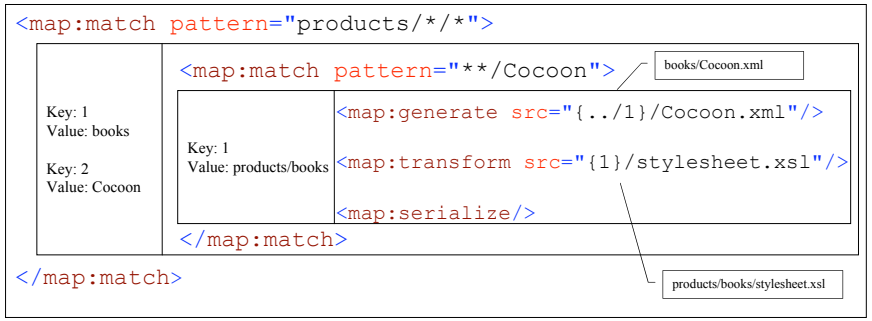
Why is this possible? Because most of the people working with Cocoon have already encountered the same problems you’re having; the code is there, tried and tested and ready to use. Cocoon doesn’t have to be about doing the heavy lifting; it can be about pulling together the right building blocks with some XML configuration glue to create all manner of applications.

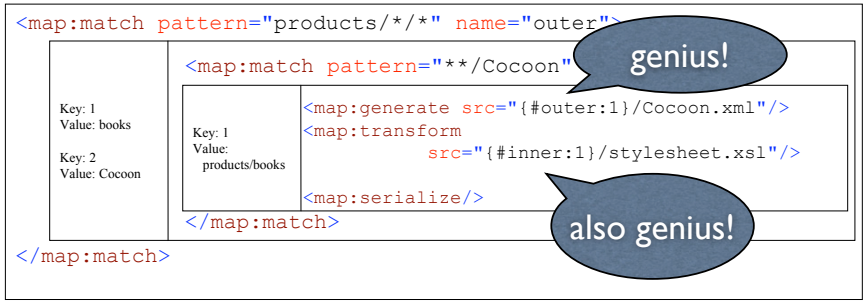
7

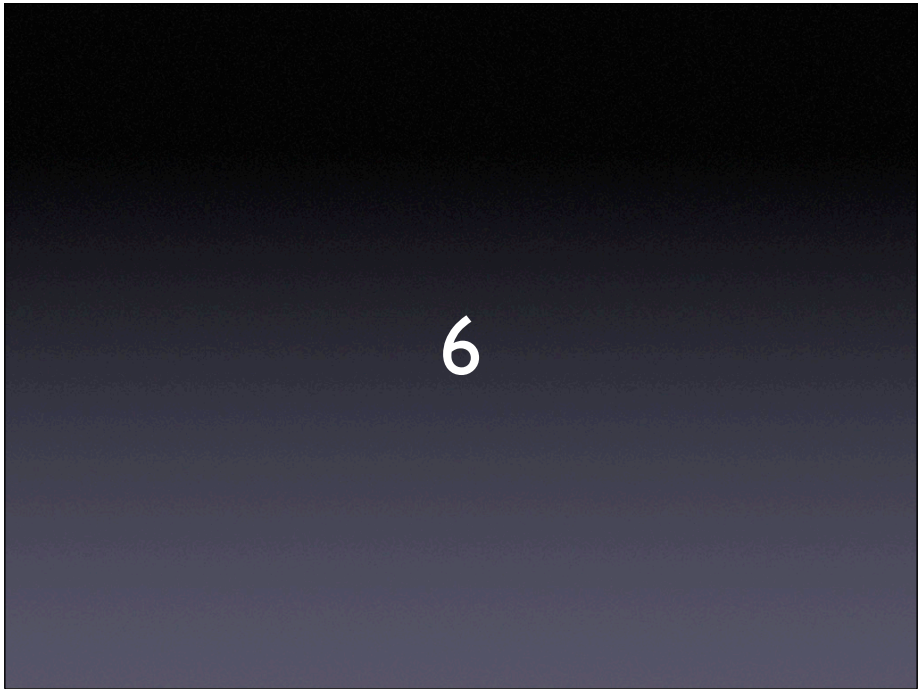


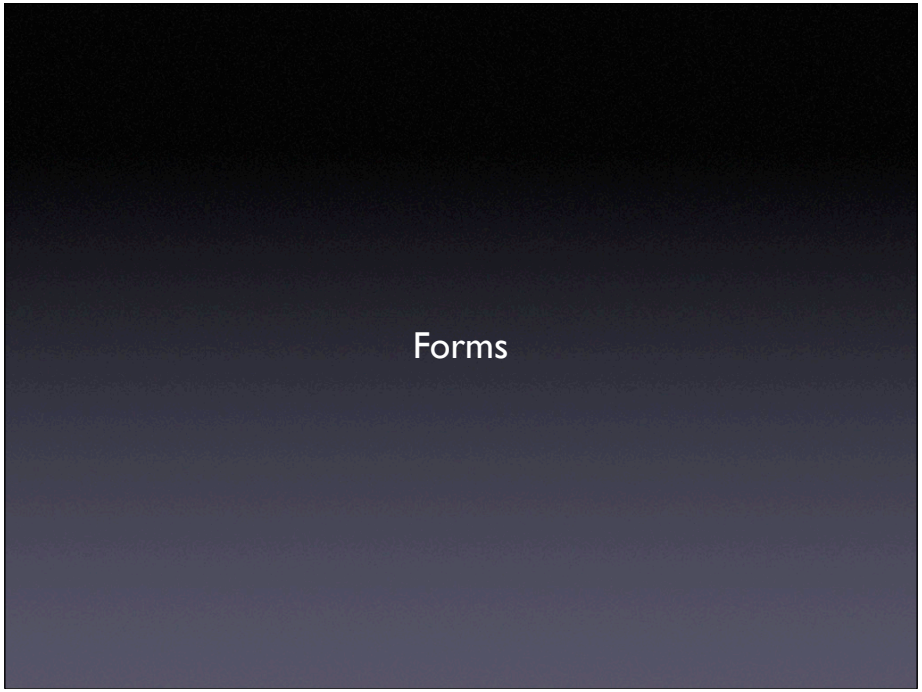
If you're new to Cocoon, the sitemap is the switchboard.
It connects up the parts of your application.
If you're an old hand at Cocoon, revisit your sitemaps and see if there isn't a way you can improve them: it's surprising the features that are in there that rarely get used.
Keep them simple -
Here's two examples:


<map:match pattern="books/*">
Key: 1
Value: Cocoon
<map:generate src="{1}.xml"/>
...
<map:serialize/>
Cocoon.xml
</map:match>

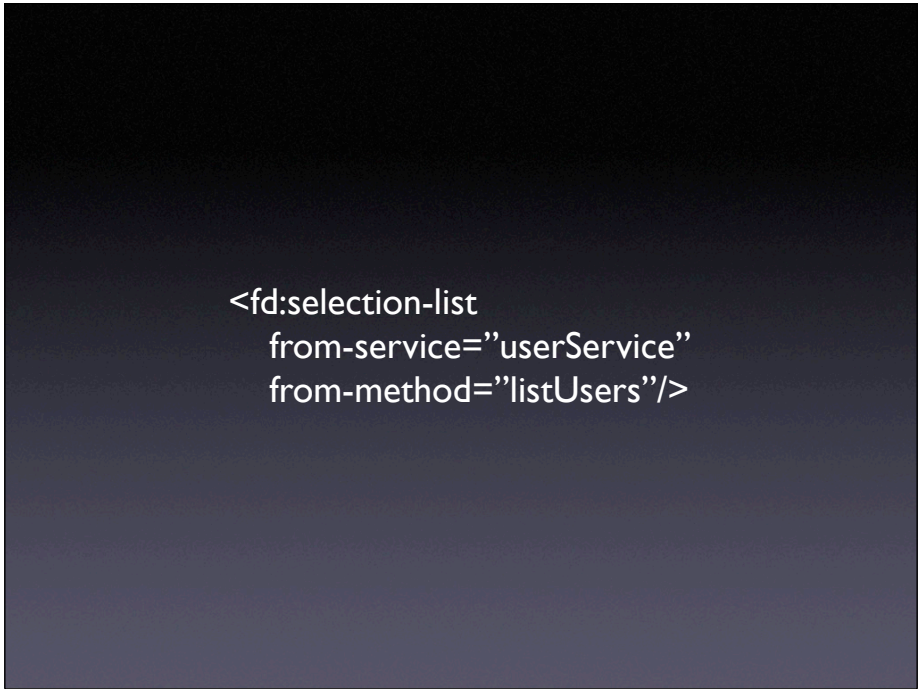

<map:match pattern="products/*/*">
Key: 1
Value: books
Key: 2
Value: Cocoon
<map:match pattern="**/Cocoon">
books/Cocoon.xml
Key: 1
Value: products/books
<map:generate src="{../1}/Cocoon.xml"/>
<map:transform src="{1}/stylesheet.xsl"/>
<map:serialize/>
</map:match>
</map:match>
products/books/stylesheet.xsl


<map:match pattern="products/*/*" name="outer">
Key: 1
Value: books
Key: 2
Value: Cocoon
<map:match pattern="**/Cocoon"
genius!
Key: 1
Value:
products/books
<map:generate src="{#outer:1}/Cocoon.xml"/>
<map:transform
src="{#inner:1}/stylesheet.xsl"/>
<map:serialize/>
also genius!
</map:match>
</map:match>


6



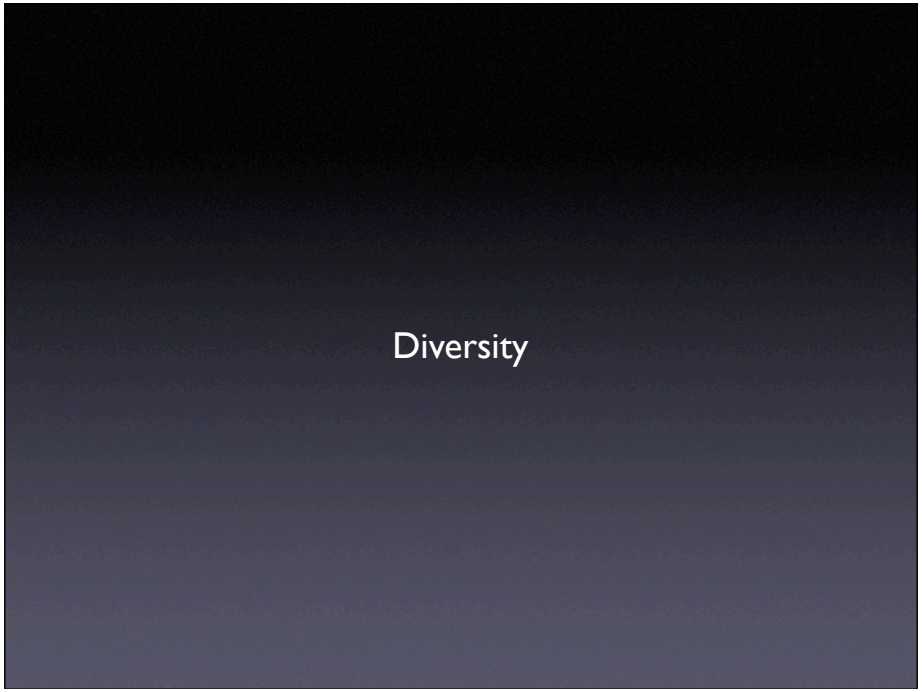
Automatic validation; complex things such as repeaters; ”just works”



<fd:selection-list from-service="userService" from-method="listUsers"/>

Set of components for cforms, configured to work directly with spring/orm layer, bypassing flow:
<fd:selection-list from-service="userService" from-method="listUsers"/>
speeds up development of large sites, keeps structure and dependencies cleaner

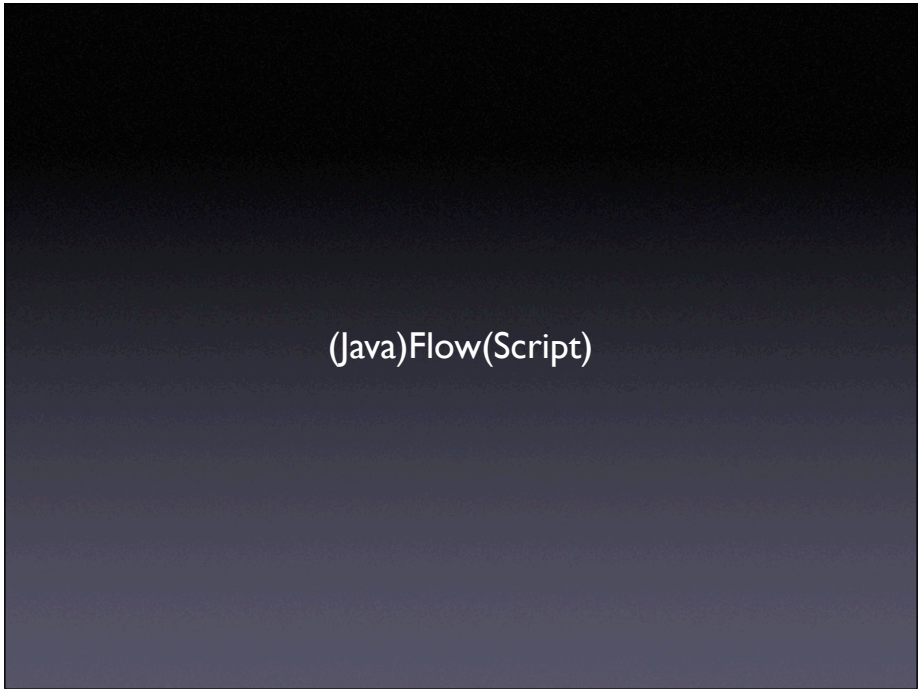
5



Cocoon used in so many different places for so many different purposes. From mobile phone networks to banks, universities to media, it's as broad as it is big.
Who can forget music in Cocoon, Midi?

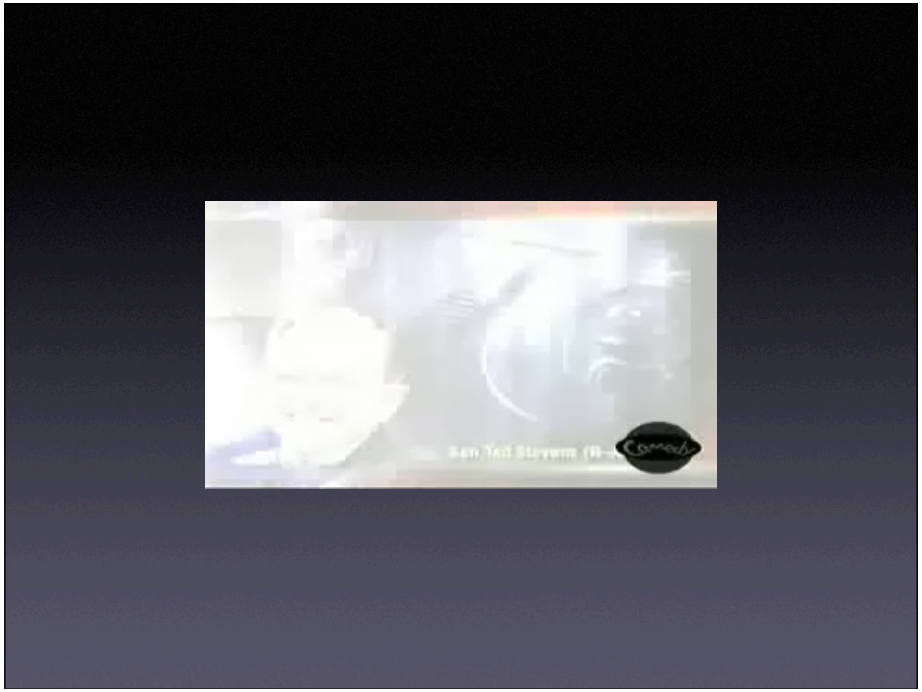
What other developer community gives you this?

4



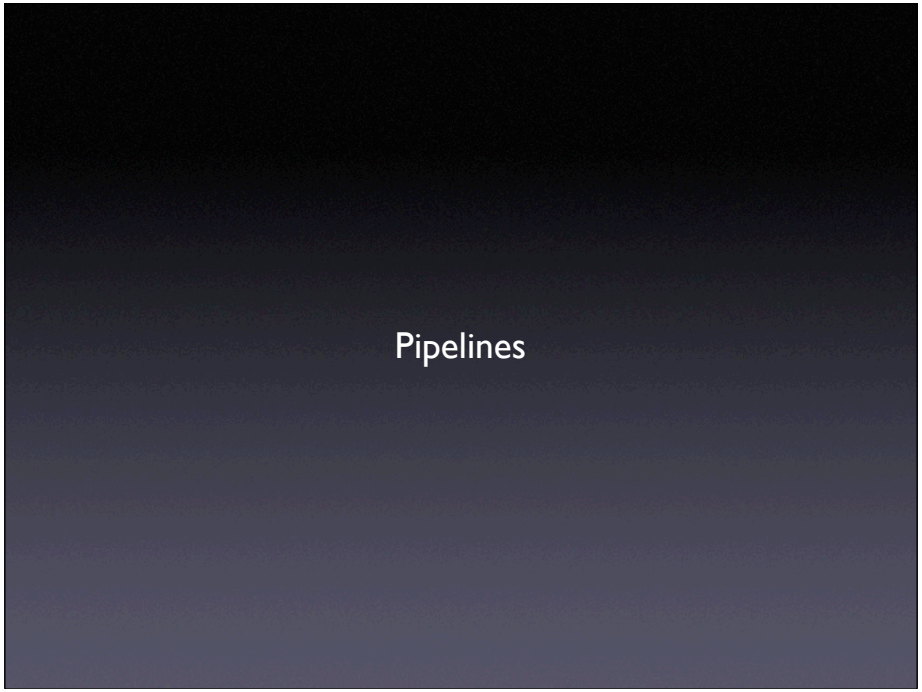
Whether you want rapid prototyping or to take advantage of continuations, flow is the answer.
<fd:selection-list from-service="userService" from-method="listUsers"/>
speeds up development of large sites, keeps structure and dependencies cleaner

how many of you are familiar with the net neutrality bill? Perhaps you caught Senator Ted Stevens (Republican, Alaska) explaining why he voted against it, with an explanation of how the internet works.



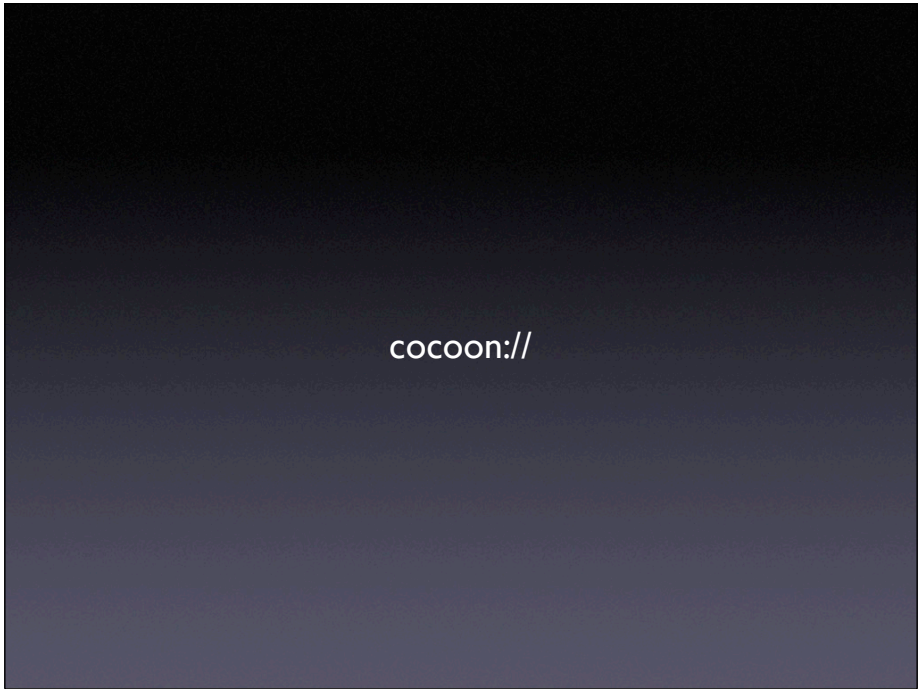
The Internet is not something you just dump something on ..... it's a series of tubes.

The poor guy got slated for this, but what most people don't know is he's actually a Cocoon developer in his spare time. He was a bit confused about terminology; what he referred to as tubes we call ...



Everything a pipeline:

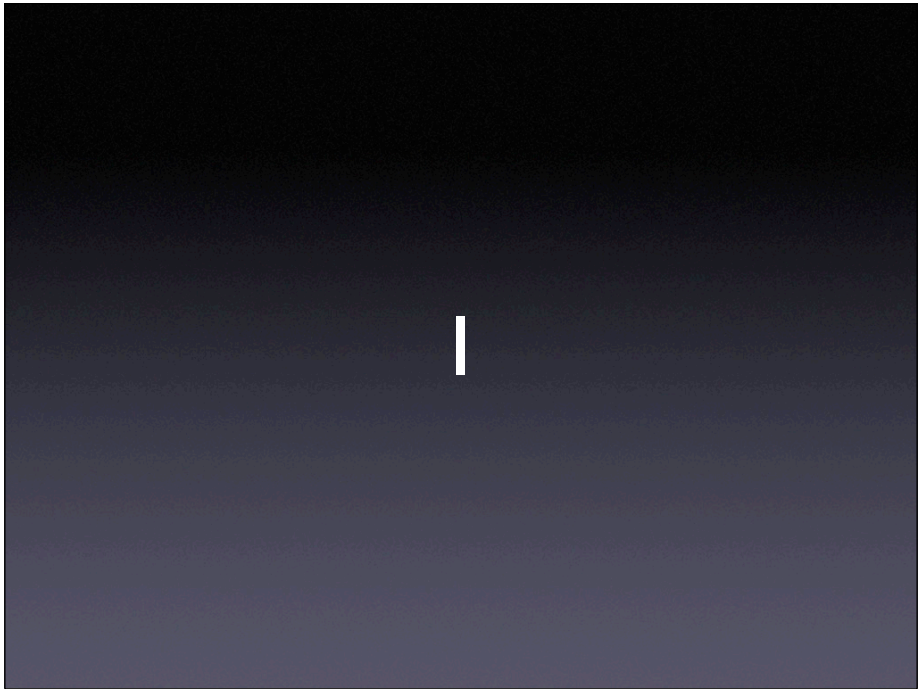
even sitemaps
even transformations



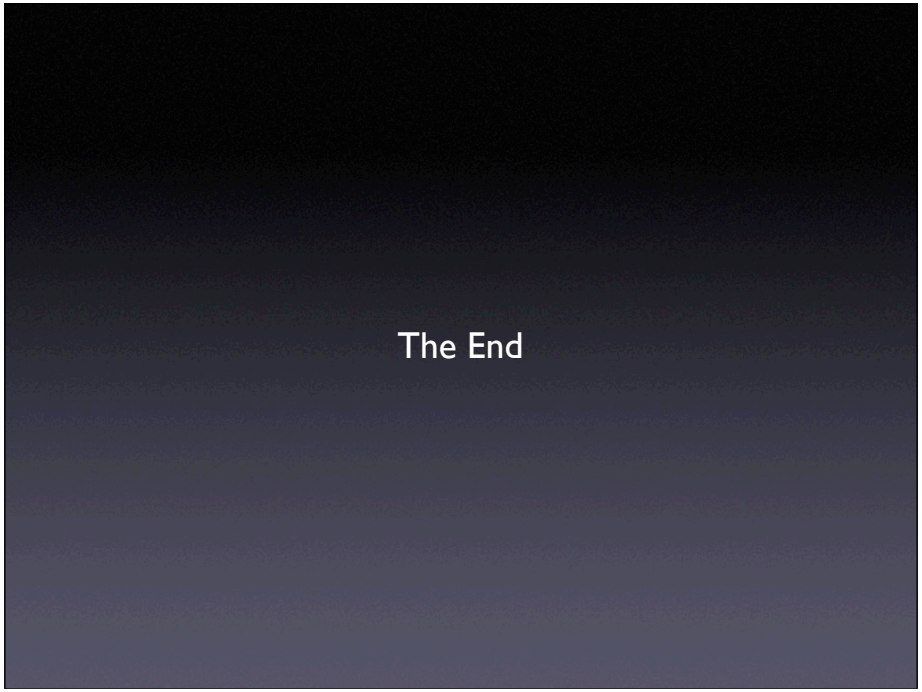
Split a complex problem into smaller re-usable bits.

2

Microsoft

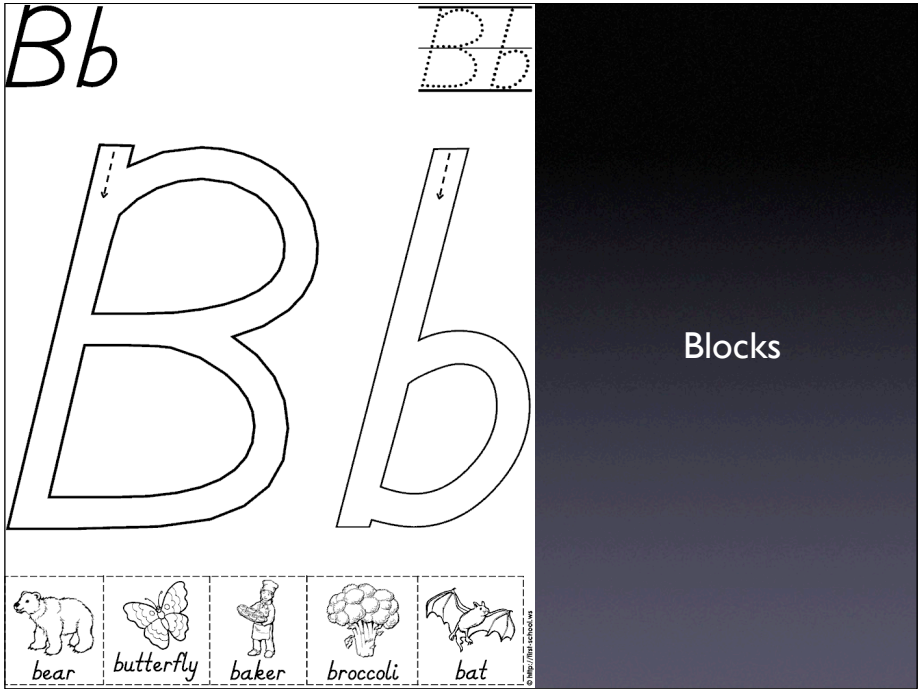


The community

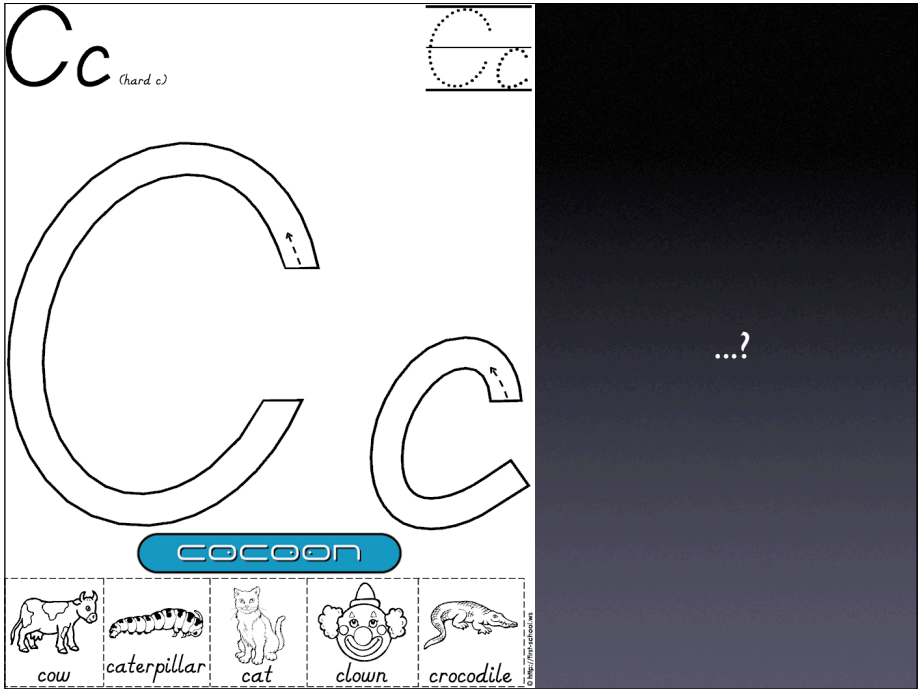


Well, time went by a little quicker than I’d expected....

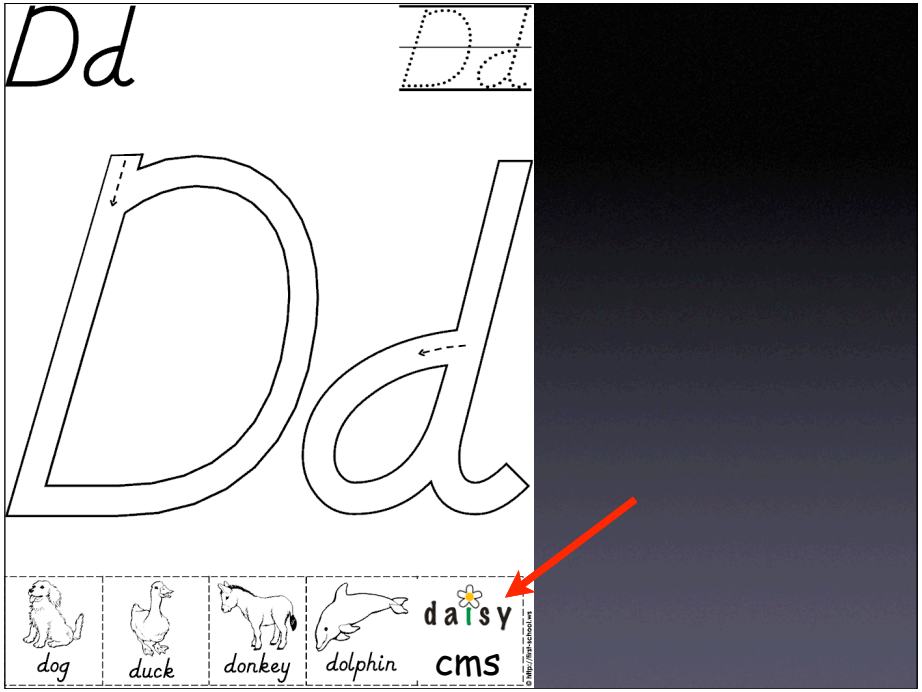
A is for ...
Actions. But it's no coincidence A is also for Awful. Use flow instead.
A is also for AJAX, an awesomely accessible addition.
A of course is also for Amsterdam.
Ok, you get the idea. Let's see if you're still awake after lunch, and how much you know about Cocoon ....

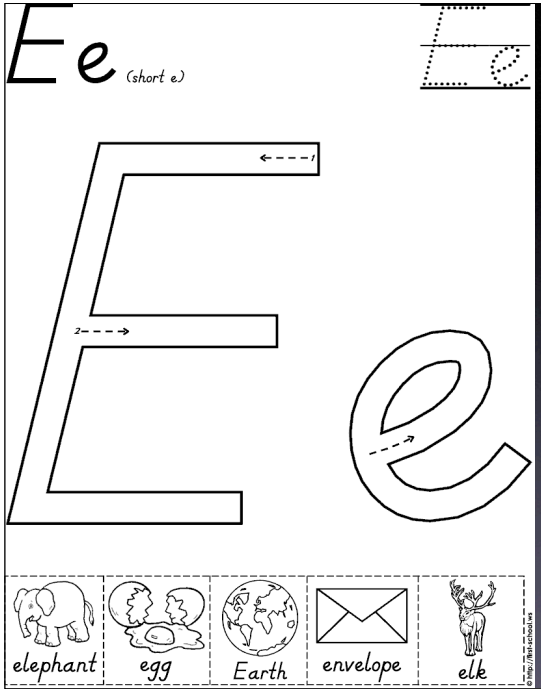


Blocks, the division of Cocoon into more specific components.
Pluggable, wireable, reloadable at runtime in Cocoon 3.

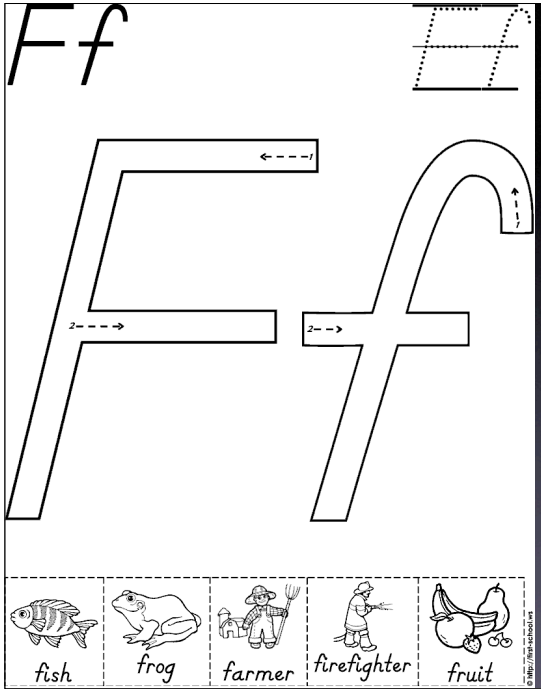


C. What begins with... ? I really struggled with this one. Sorry.


Dd
Dd
Dd
dog
duck
donkey
dolphin
daisy
cms

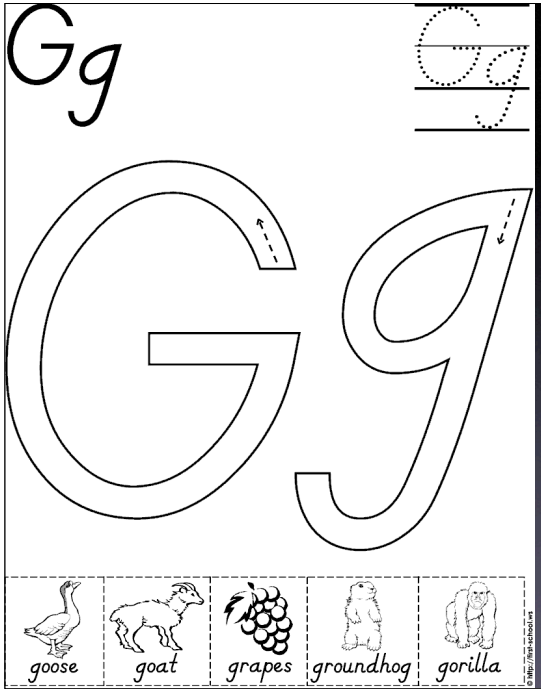


ehcache

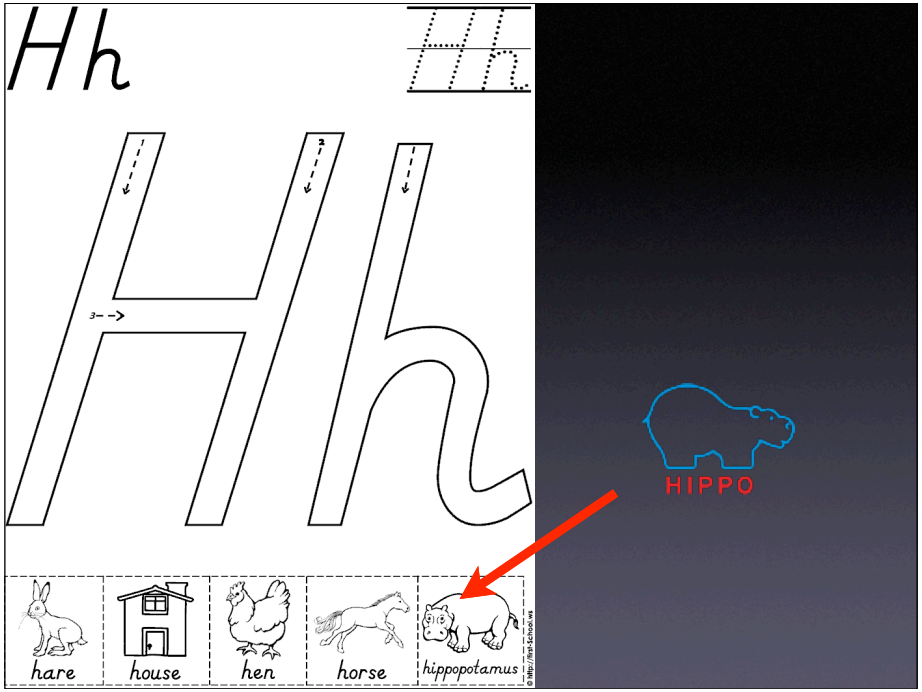


Flow

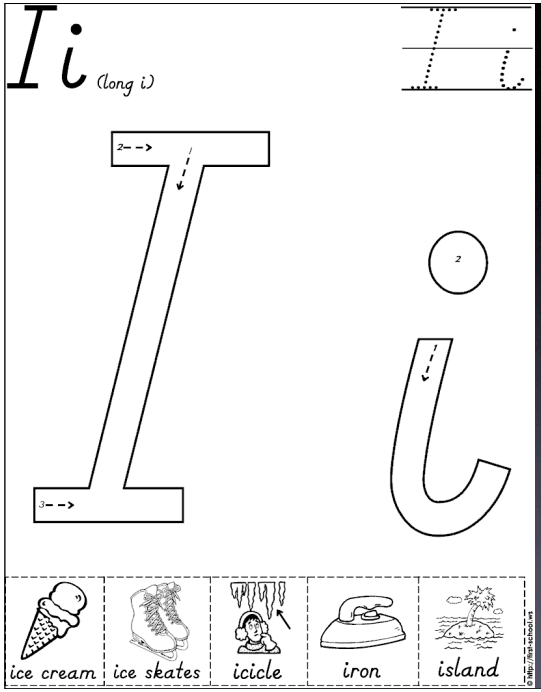
Forms



Generators

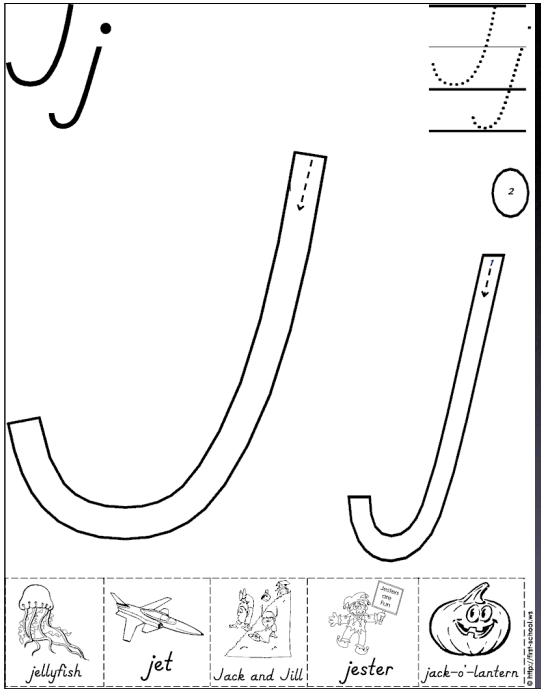


Look! Look! It’s a hippo! What a bit of luck!

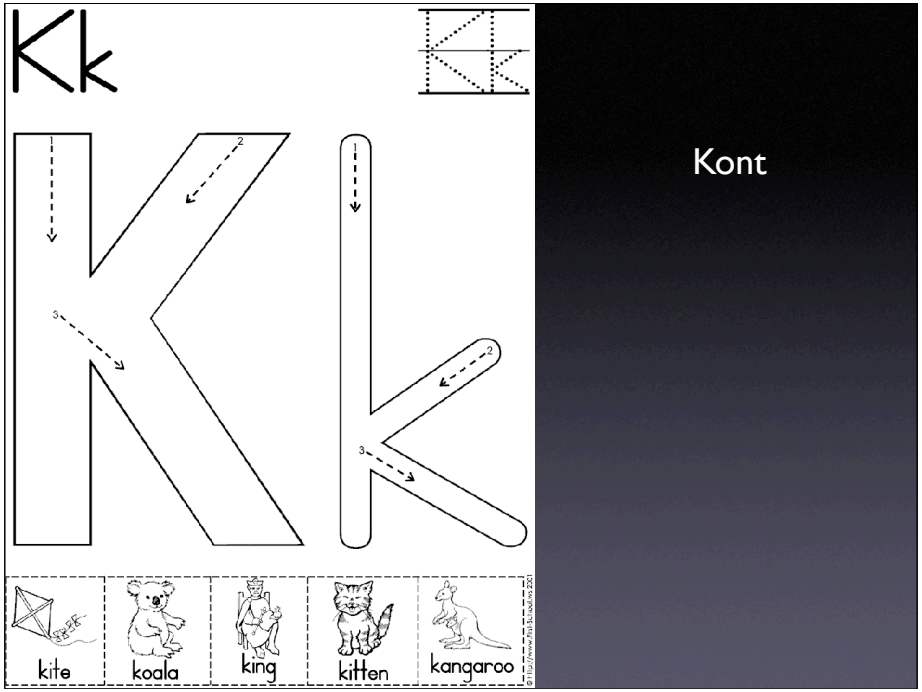


i18n

Integration



JavaFlow



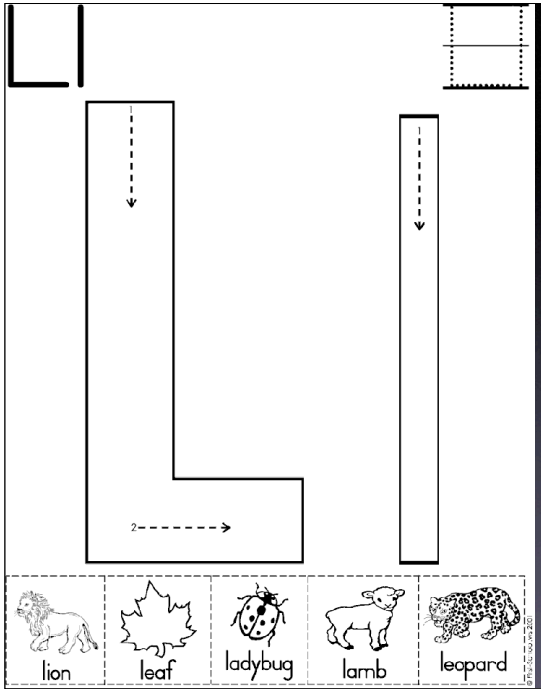
I'll help you out with this one.

K is for 'kont'... a german abbreviation of "continuation".
Unfortunately, it's also dutch for "arse".
Moral of the story: be careful with your URLs.

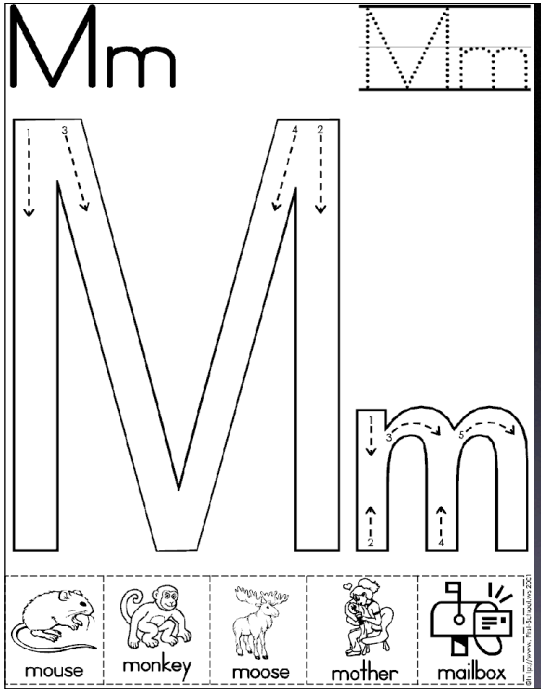
K is for ‘kont’... a german abbreviation of “continuation”.
Unfortunately, it’s also dutch for “arse”.
Moral of the story: be careful with your URLs.

```
<map:match pattern="*.kont">
<form action="${continuation.id}.kont" method="POST">


    <map:match
        type="request-parameter"
        pattern="continuation-id">
        <map:call continuation="{1}"/>
    </map:match>
    <ft:form-template action="" method="POST">
    <ft:continuation-id/>
```

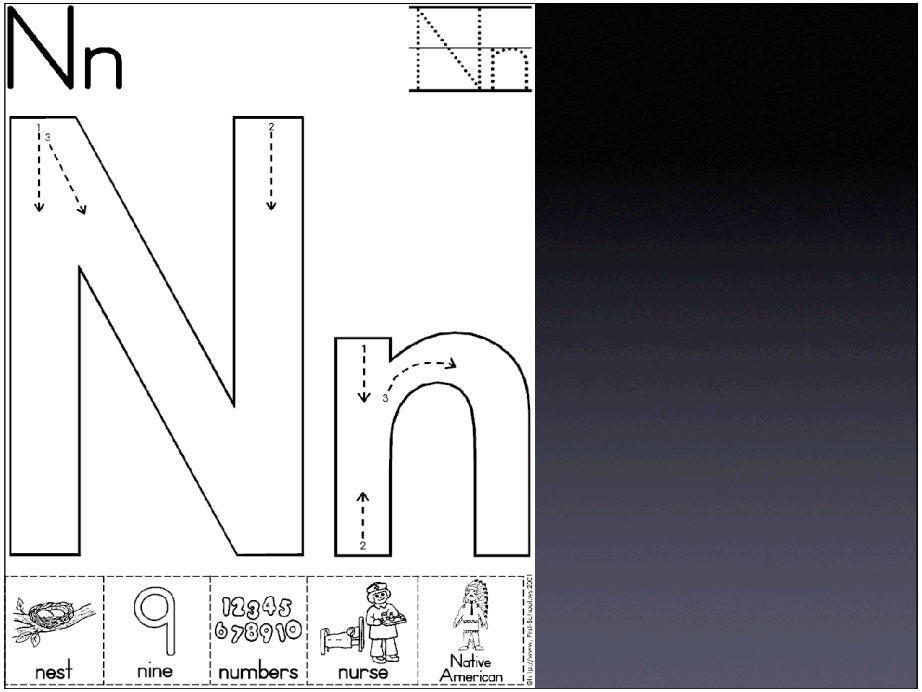


110n

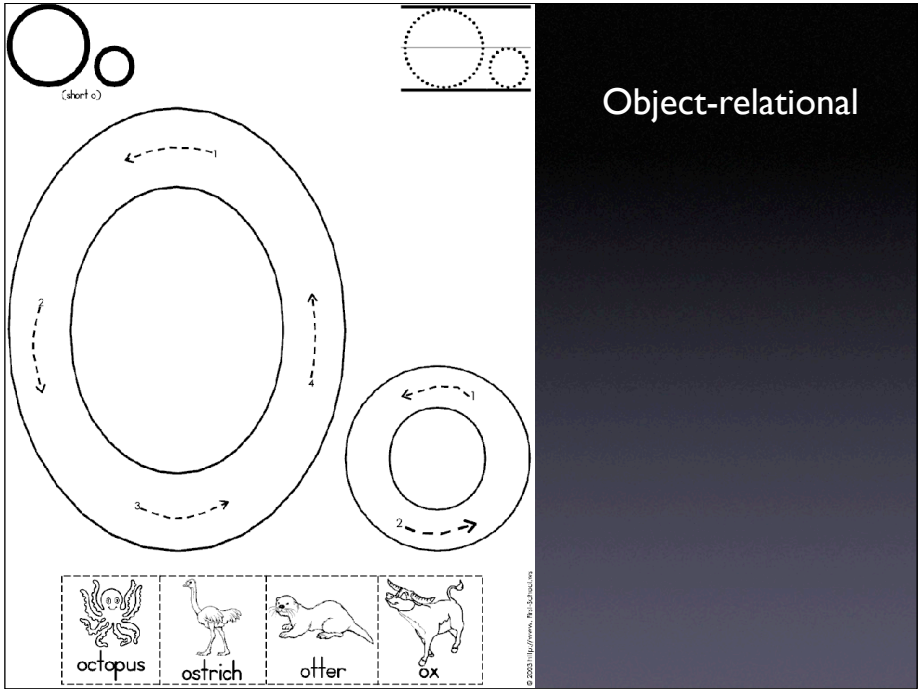


Multichannel

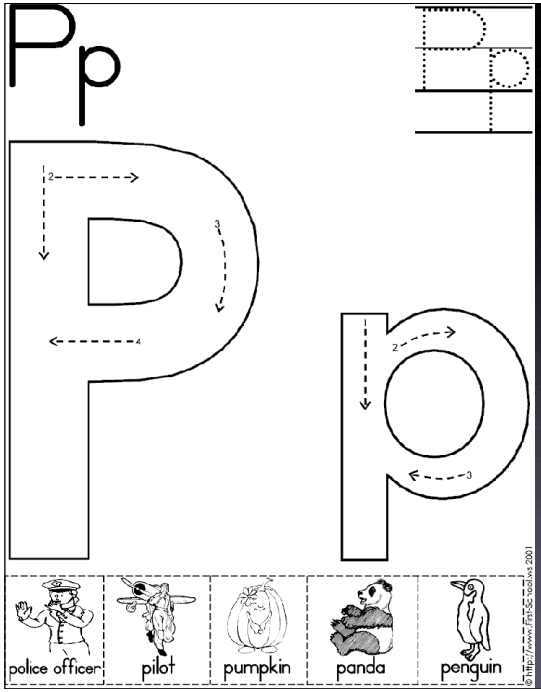
Mavenisation



naming

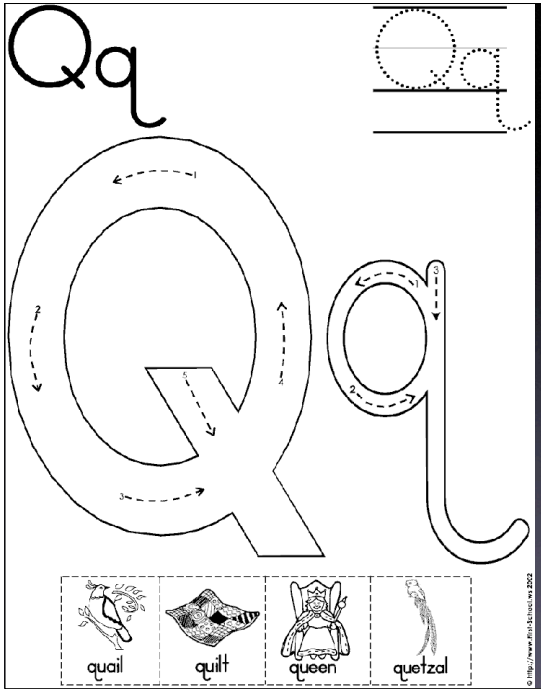


... is for “Oh my goodness is he really going to read out the whole alphabet?”

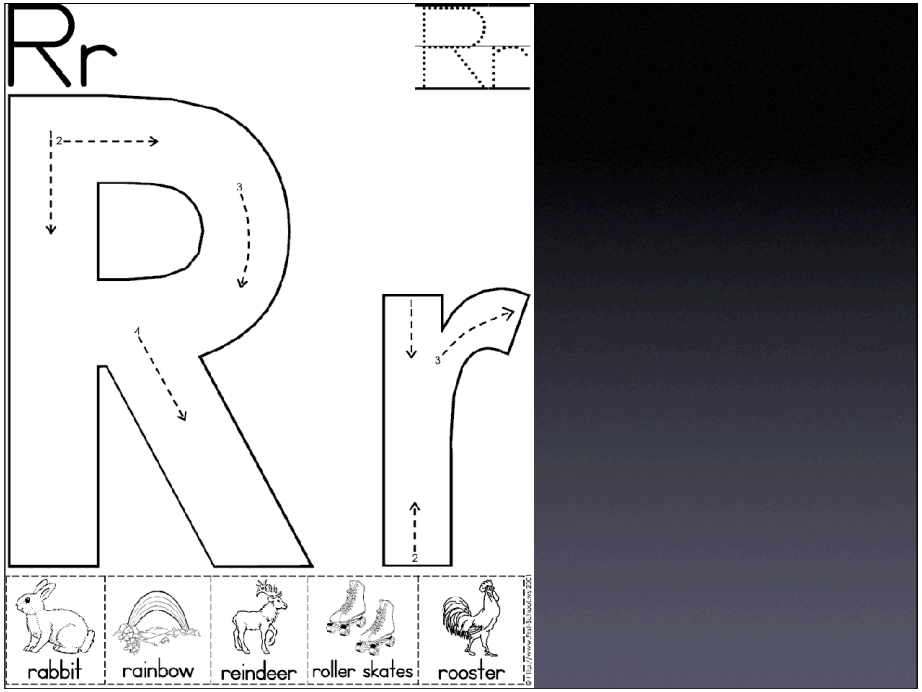


Portals

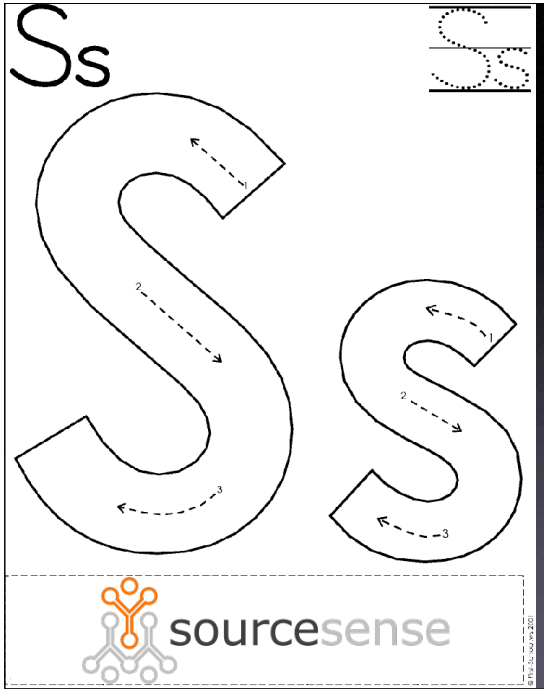
... is for Perform.



Querybean

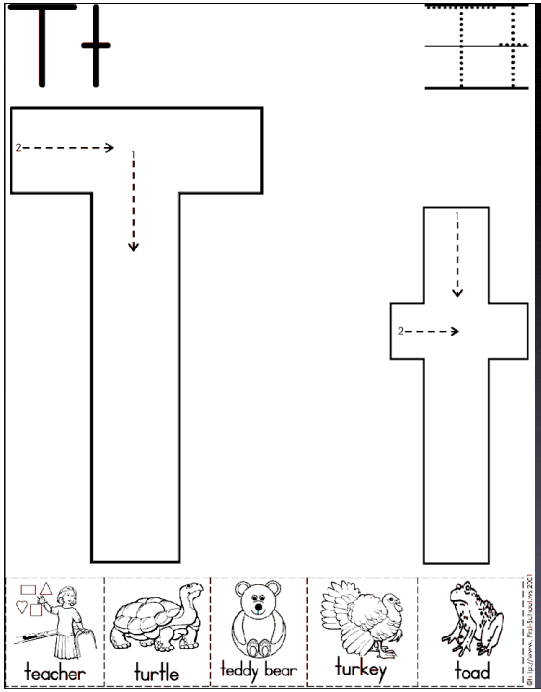


repository
readers

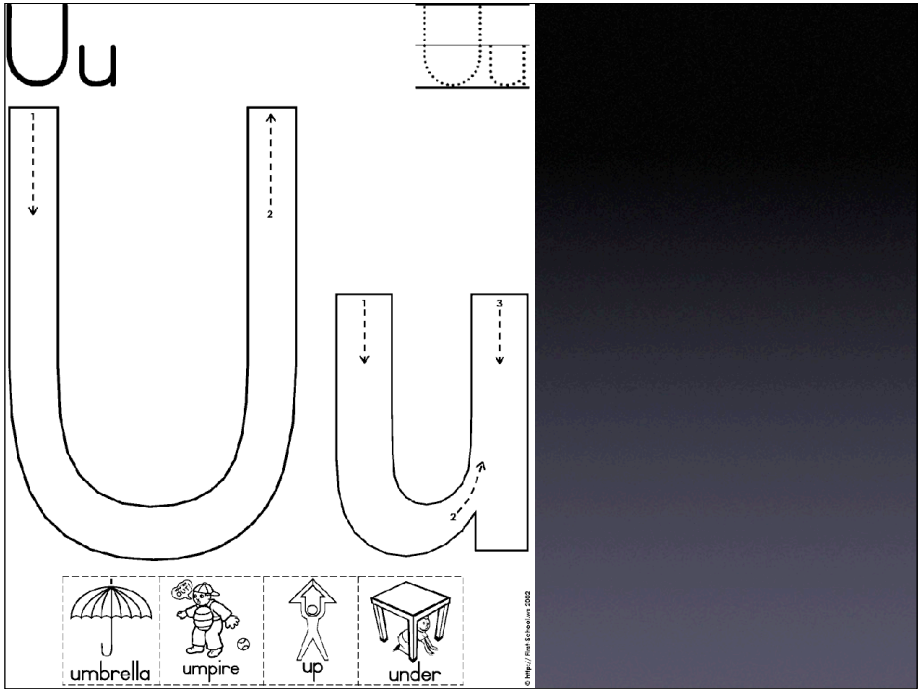


Sitemaps

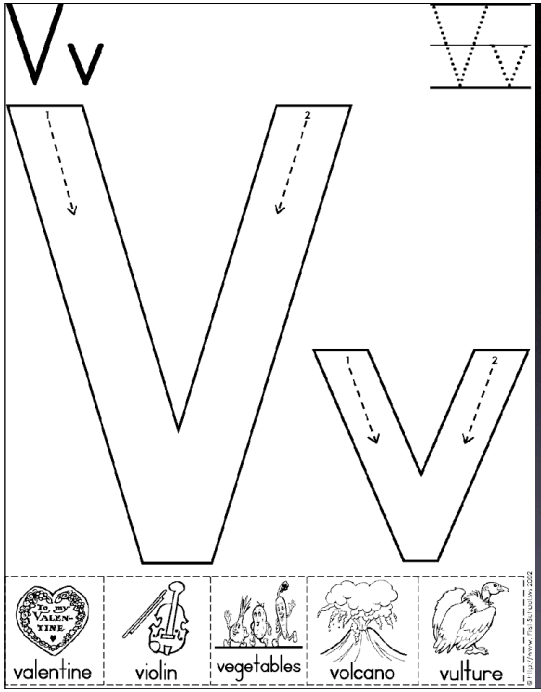
Serializers



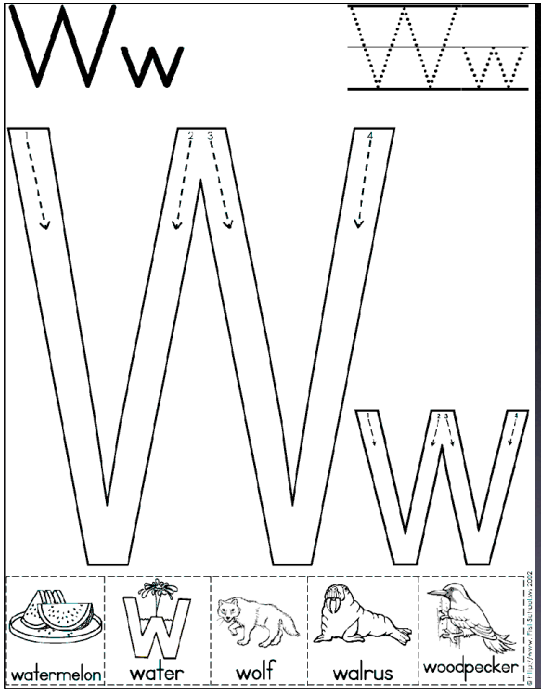
Transformers



util

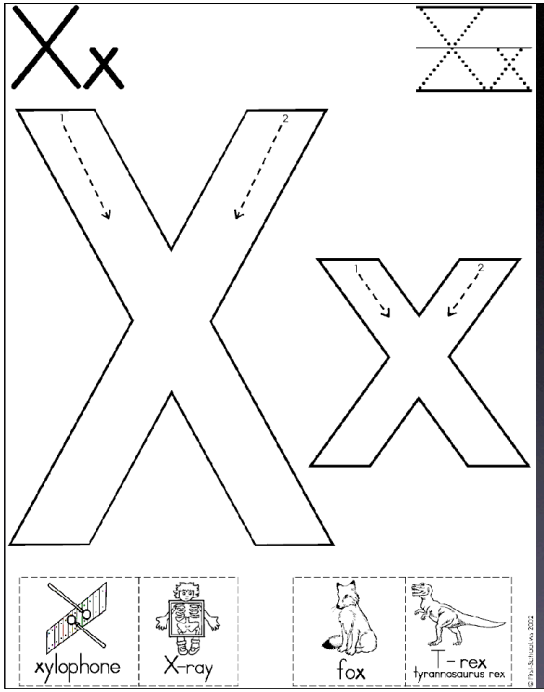


<Verbose></Verbose>

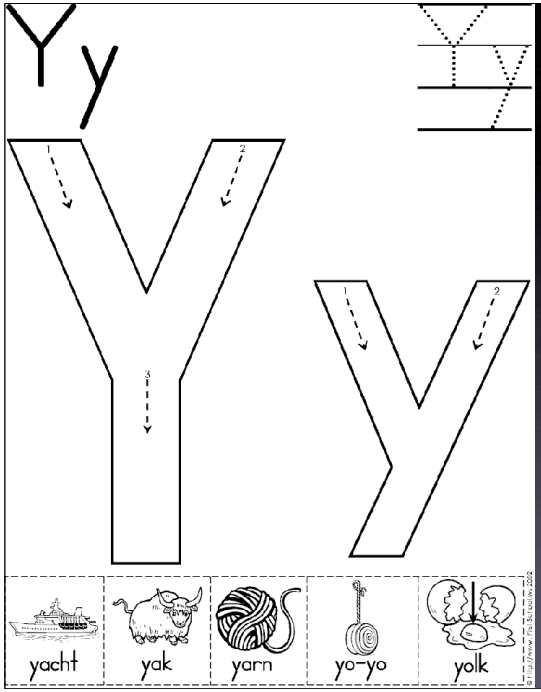


Web 2.0

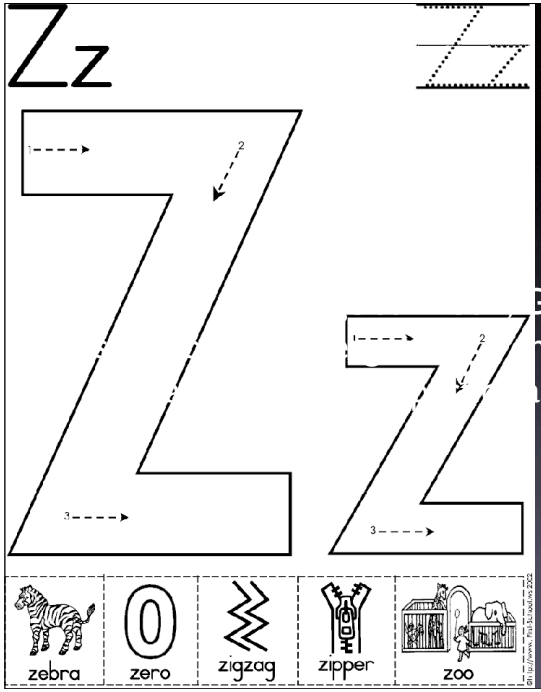
Why aren't you using Cocoon?



<?xml version="1.0"?>

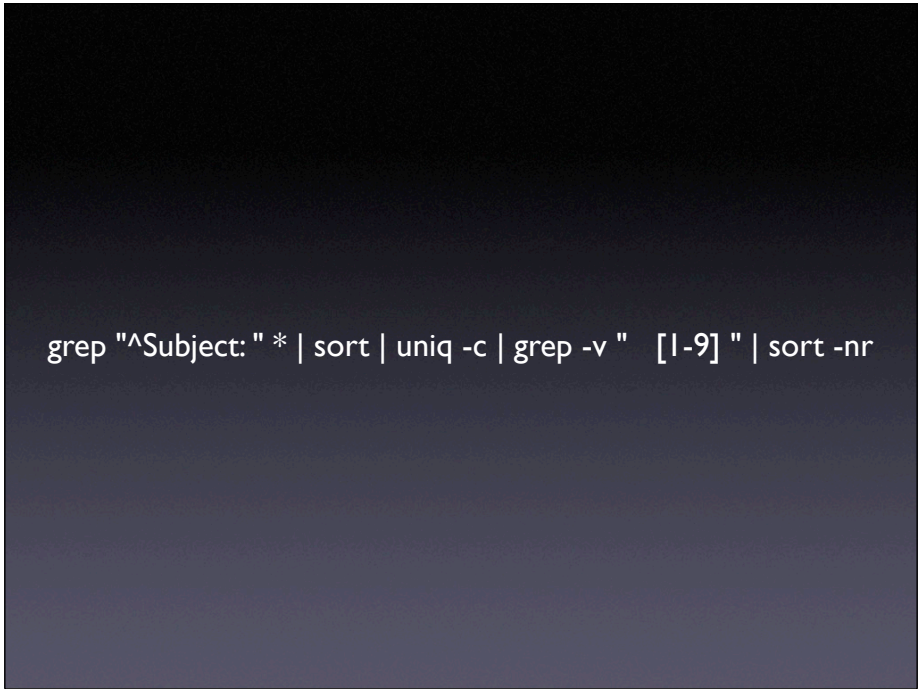


you.



Zeitgeist

German.
nd of thought or feeling
r period of time.



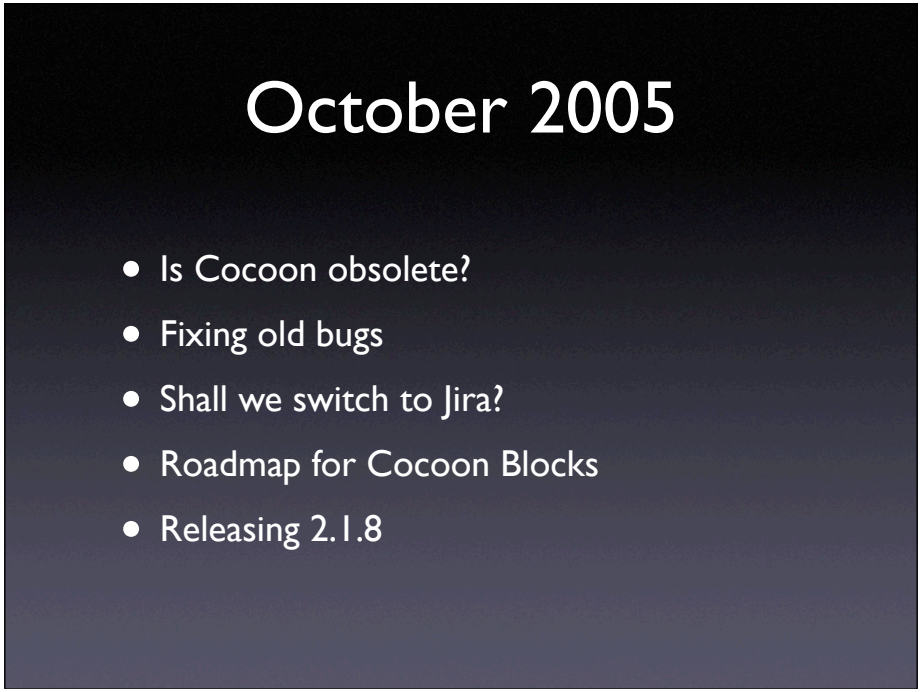
Calculated using advanced heuristics, natural language processing, lexical patterns, dice coefficient, latent semantic analysis

a hacked together shell script (yes I was hacking on *something* at the hackathon)

Partly a trip down memory lane, partly a look at the history of the last year to seriously identify common themes, problems and successes.

Here's a look at the most popular themes of September

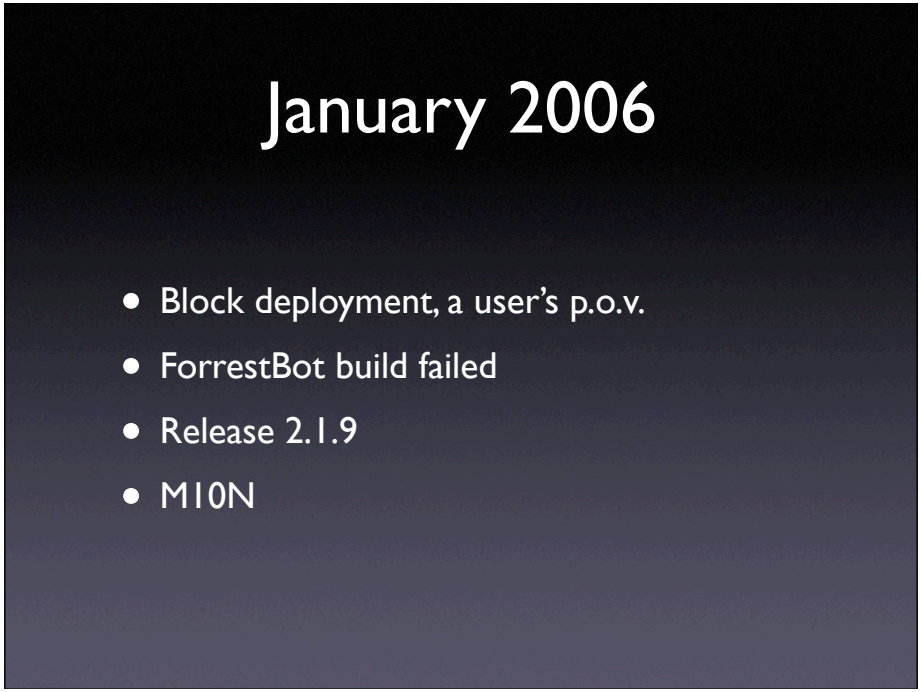
The buzz back in September surrounded Jira, a GUI, Arjé and 2.1.8



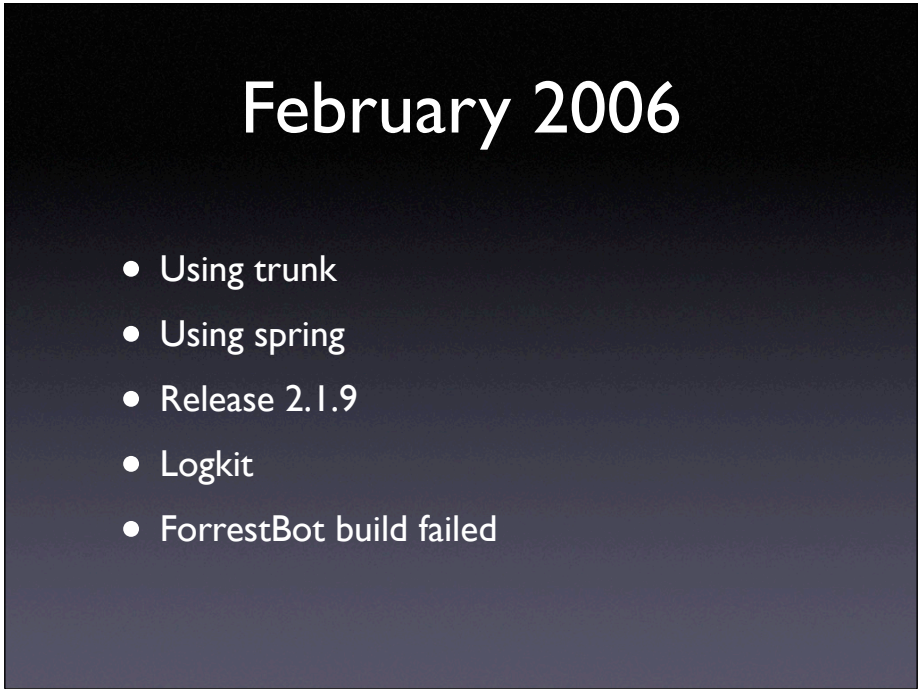
In October, the anti-buzz was on obsolescence, old bugs, Jira, blocks, and 2.1.8

In November, we looked to 2.2, ForrestBot was complaining, we talked about logos, and 2.1.8
Oh, and M10N started entering the charts ...

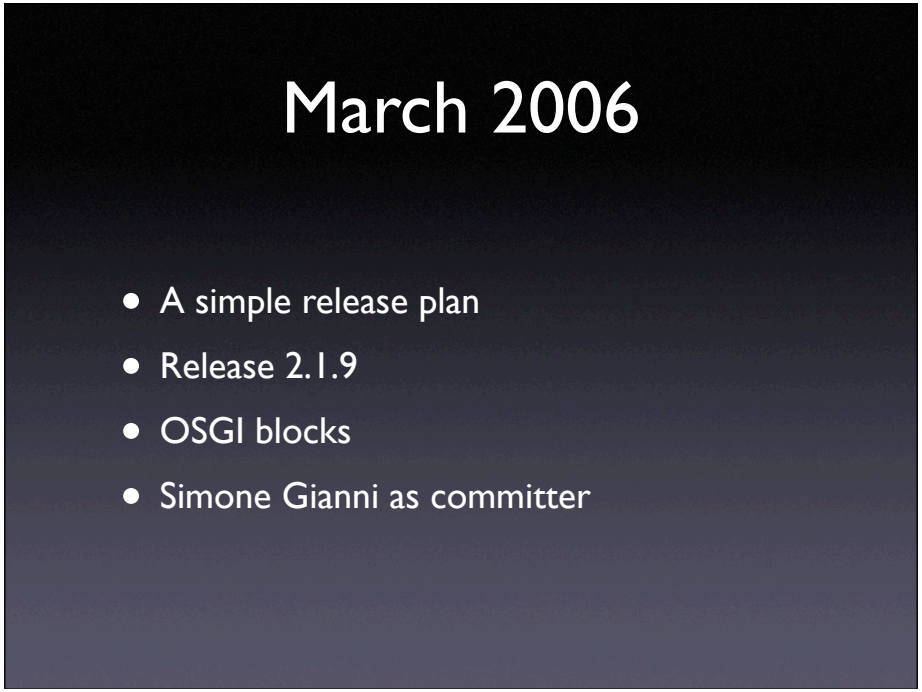
In December, flush with our victory of 2.1.8, we started talking about Cocoon 3, shiny things, Maven, and AJAX. ForrestBot would not shut up.



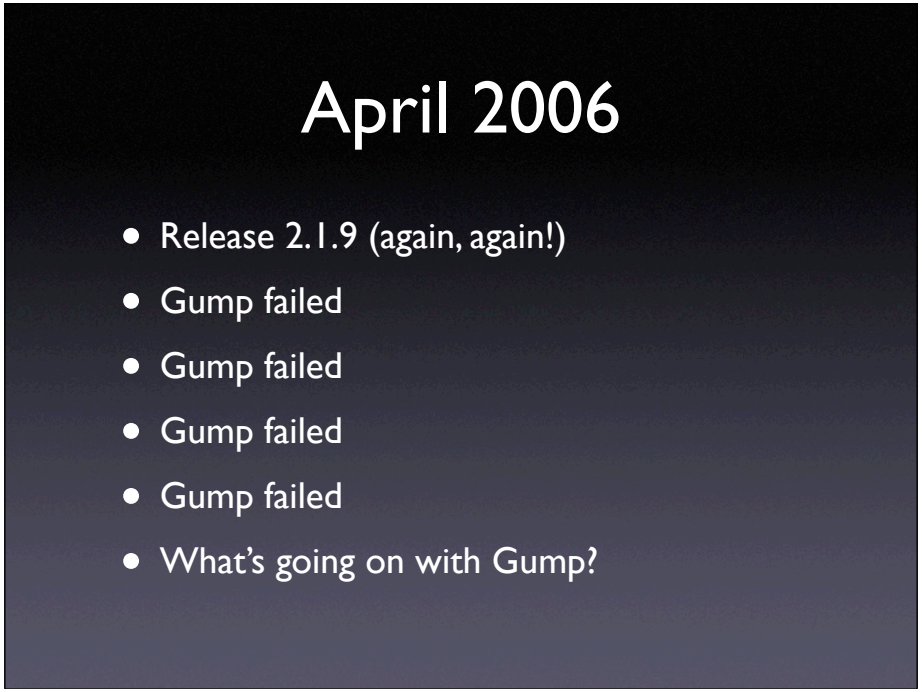
Blocks, would someone have a quiet word with ForrestBot, 2.1.9, M10N



February was user month, with talk about using trunk and spring. 2.1.9. Logkit. ForrestBot.



March saw us come up with a cunning release plan, and talk about 2.1.9
OSGI got popular, so did Simone.
... and just 6 short months later, he gives us blueprint!



2.1.9 for third month running.
Gump got jealous of ForrestBot's success.

ForrestBot lives!

June 2006

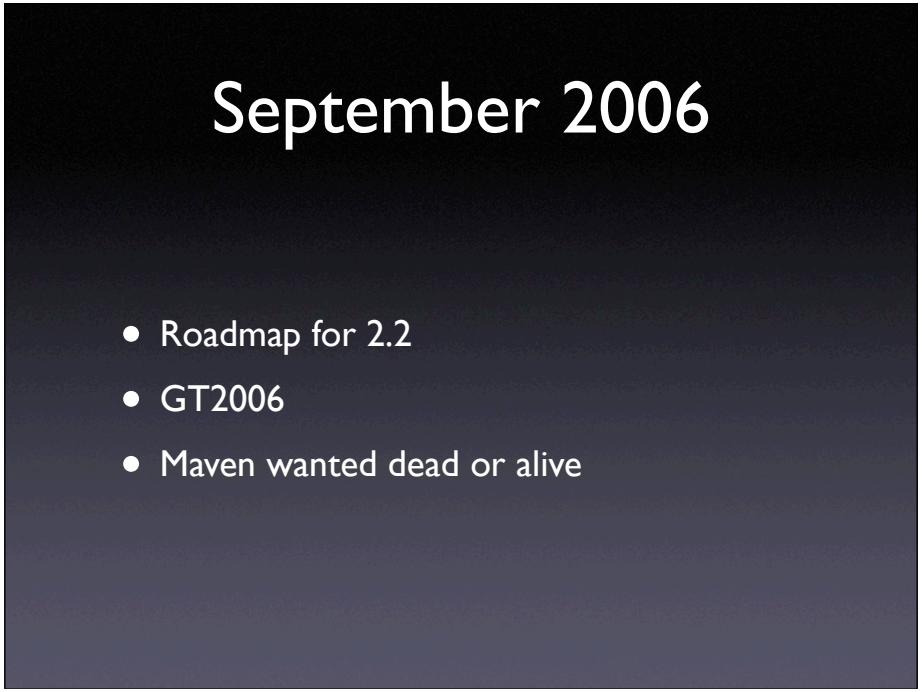
- ..... holidays?
- Andreas, Peter and Jason as committer

July 2006

- Ard Schrijvers as committer
- New processor interface
- Release 2.2m1
- Continuations consuming RAM
- Maven

August 2006

- Java 5 minimum JDK requirement
- Cocoon 2.2 and Java 5
- Release 2.2m1
- ForrestBot build failed
- Who is this Java 5 anyway?



western sound clip?

The End
(really, this time)