Design Pattern Safari

Peter J. Farrell
GreatBizTools, LLC

About Your Guide

- Co-host of the ColdFusion Weekly
- Lead Developer for Mach-II
- CIO of GreatBizTools, LLC
- Classically trained musician

Our Safari's Itinerary

- What is a Bean?
- What is a DAO?
- What is a Gateway?
- What is a Service?
- How do they all work togeather?

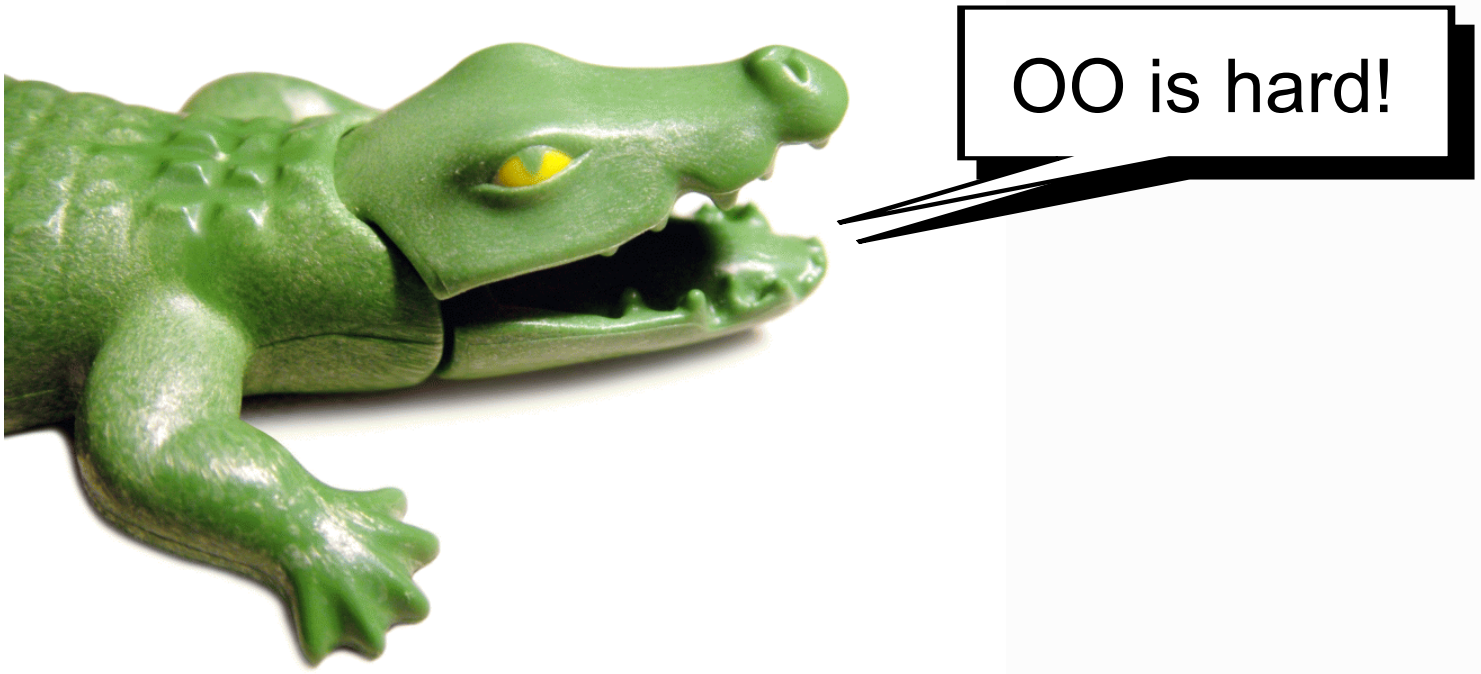
Part One
Embarking On The Safari

What About My Passengers?

- How many have used CFCs before?
- How many have at least some OOP experience with either ColdFusion or other languages?
- How many use a framework such as fusebox, Mach-II, Model-Glue?

Safety First! – Ask Your Guide Questions

- **Safari's can be dangerous.**
- **Do not get eaten by an object!**



The Objectazon River

- A design pattern is a general and repeatable solution to a commonly occurring problem
- A design pattern is not code, but a structure or blueprint to implement your code by
- A design pattern is not an algorithm since algorithms solve computational problems

Starting the Safari
Why Patterns Are Important...

- Patterns provide a “standardized” vocabulary for developers to communicate using well-known, well-understood names for software interactions
- Design patterns can easily be improved over time, making them more robust than ad-hoc designs
- Patterns are about extensibility and reusability

Part Two

Comon Indentifications In the Jungle

- Beans
  - AKA Value Objects
- DAOs
  - Data Access Objects
- Gateways
  - Data Gateway Objects
- Services
  - AKA Manager Objects

Beans - Part 1

Comsume these for recommended daily data intake

- Typically represents a specific entity in your model
- Carries “encapsulated” data between the different layers of your application
- Has a consistent and simple interface

Beans - Part 2

- Has methods called getters/setters (aka accessors) to access data
  - getFirstName() / setFirstName()
- Might be composed of other beans
- Easily created with a code generator
- **Let's see some code...**

DAOs

Your Access to All Things Single

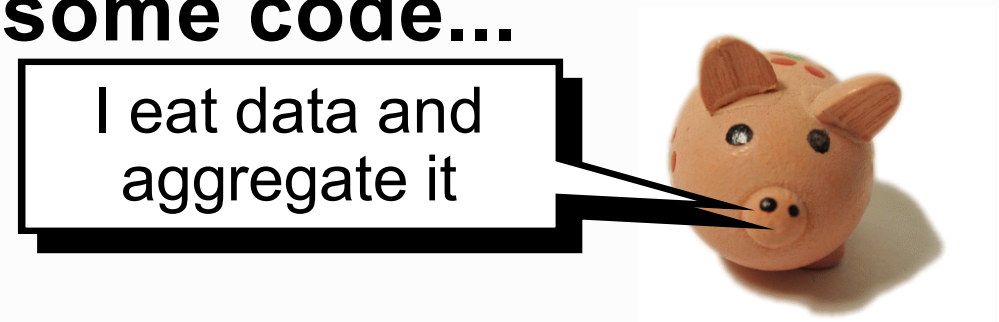
- DAOs only interact with one row of data via the primary key
- Used to save/load objects from data storage
- A DAO could interface with:
  - Database / Legacy persistent data storage
  - XML / Text File

DAOs - Part 2

- My DAOs usually have ScRuD method that take a bean
  - **S**ave | (**c**reate) | **R**ead | (**u**pdate) | **D**elete
- **Let's look at some code...**

Gateways

- Contains method that sql (select) queries that return one or more rows of data
- Typically returns a cfquery object
  - Could return a bean (Querying by User/Pwd)
- Performs functionality that may affect one or more rows
- **Let's look at some code...**



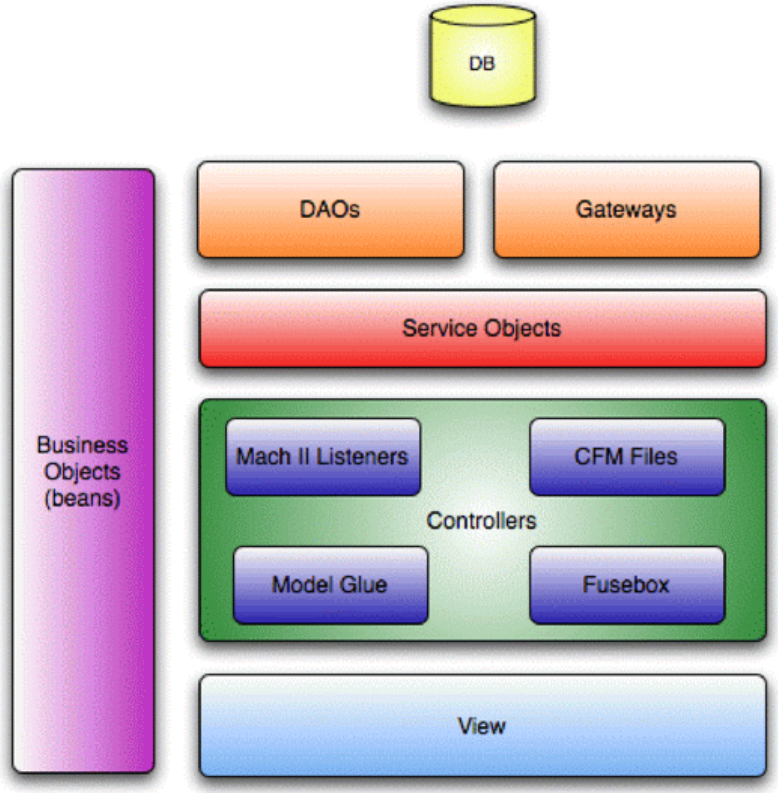
Services - Part One
Kings of the Jungles

- Contains your business logic:
  - Bean Validation / Creation
  - Application specific business logic
  - Save / Load / Delete from DAO (via the facade pattern)
  - Gateway Interaction (via the facade pattern)
- Usually depends on a DAO and Gateway

Services - Part Two

- Used by your application via:
  - Controller (frameworks) / Action Page
  - Web Services / Flash Remoting
- Services can comsume other services
- Provides a single point of entry
- **Let's look at some code...**

Part Three
Tying It All Together



Others Notable Stops in the Wild
Other Design Patterns You Should Know About

- Facade
  - Provides a simplified interface to a larger body of code
- Controllers
  - These are very framework specific. Recommend looking at Fusebox, Mach-II or Model-Glue.
- Singletons
  - An object that only has one instance during the lifetime of the application.

Obey Your Survival Instincts

- Use design patterns only when you understand the problem. Don't throw patterns into you application without really understanding them.
- Focus only on your needs. Remember that some patterns have tradeoffs.
- When you code has a particular kind of problem or "smell", go your pattern toolbox to find a solution.

Essential Navigational Equipment

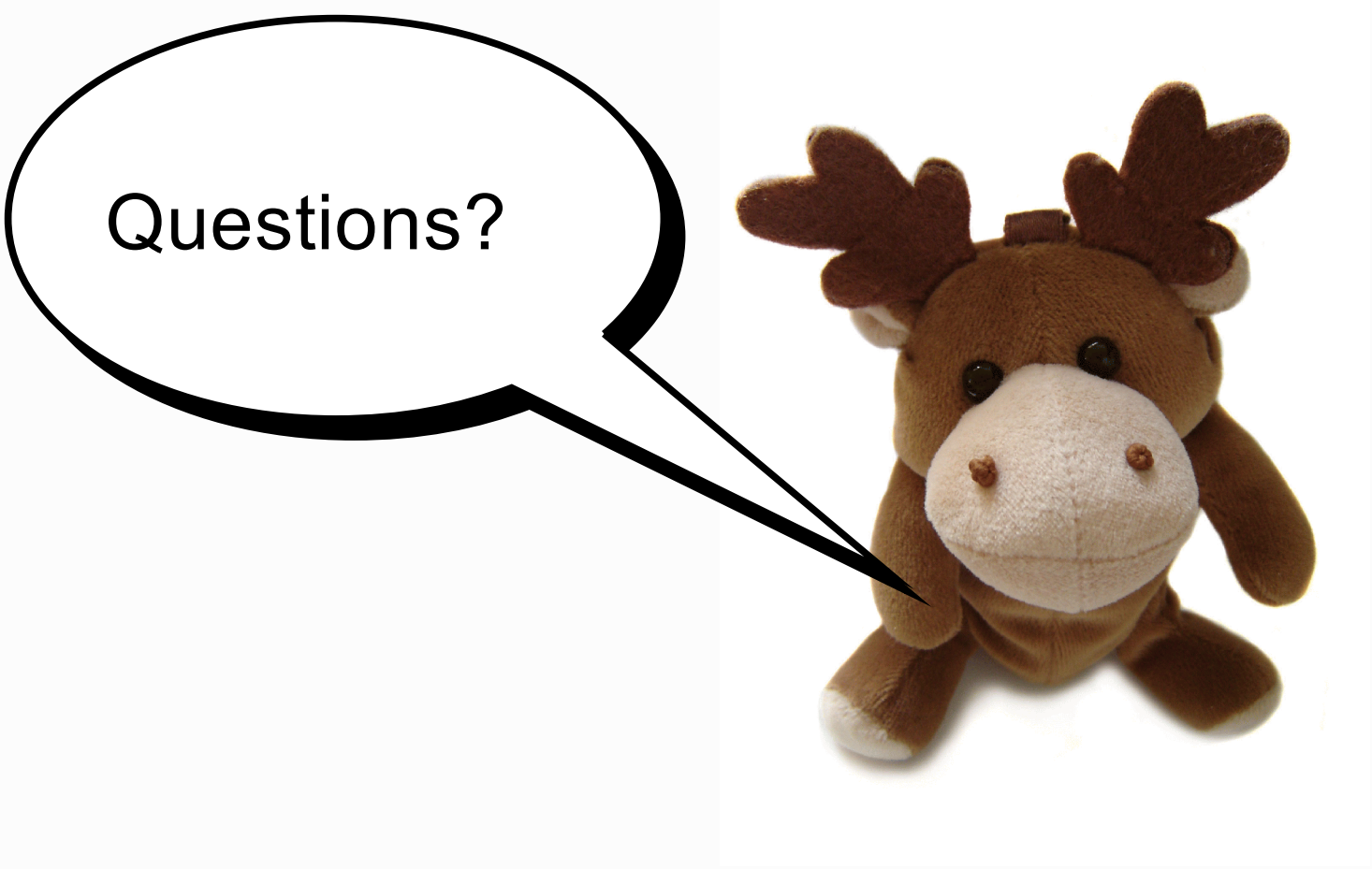
Learning, Generating and Managing Objects

- Desire to learn and willingness to hit your head against the wall many times.
- Code Generation - Rooibos Generator
  - rooibos.maestropublishing.com
- Object Management - ColdSpring
  - Focuses on making configuration and dependencies of your objects (CFCs) easier to manage. ColdSpring uses the “inversion-of-control” pattern to “wire” your CFCs together.

Become a Safari Guide

- Object Technology: A Manager's Guide
  - David A. Taylor
  - Addison-Wesley Professional – 0201309947
- Design Patterns Explained: A New Perspective on Object-Oriented Design
  - Alan Shalloway & James Trott
  - Addison-Wesley Professional – 0321247140
- Head First Design Patterns
  - Freeman, Freeman, Bates & Sierra
  - O'Reilly Media – 0596007124

Ending Our Safari



Smoke Signal Techniques

- Peter J. Farrell
  - Email: pfarrell@greatbiztools.com
  - Blog: blog.maestropublishing.com