Lecture 11: Frameworks (CakePHP and JQuery)

Model-view-controller framework
- Starts with the client (web browser)
- Sends request to server
- Dispatcher decides which controller to send request to

Address book application example:
We need
- a database for contacts (mysql? and php logic to query/update database)
- we need to ask the client/user what they want to do with the data
1. Request for all the contacts gets sent to dispatcher
2. Contacts controller for the address book (controller is the heart of the process)
3. Controller contacts model to perform necessary query based on what the user wants to do
4. Model returns data to the controller
5. Data to be displayed gets passed to view, which formats it all and sends it out for the client to view

Ruby on Rails - works similar to the model-view-controller
CakePHP is a ruby on rails for php

Most CakePHP code comes from 1.1 stable release
Has all the files to assemble the framework for you
YOU insert the logic
- must name files in a specific way for cakePHP to recognize them and use them
- but you don't have to deal with configuration files!

When you download cakePHP and set it all up, you're ready to create your application!
CakePHP can let you view and update your database automatically
- time fields are autopopulated by cake
- scaffolding automates certain actions (view, edit, delete)

THE CODE
Folders for controllers, models, views, etc.

THE CONTROLLER (STORES LOGIC BETWEEN MODEL, RECEIVES FROM DISPATCH, SENDS TO VIEW)
- class AppController (you build your application upon, already has many methods)

```
class AddressesController extends AppController {
     // explicitely define the name of class
     var $name = "Addresses";

     // set up a scaffold
```

```
     var $scaffold;
}
```

MODELS
- class AppModel (your class builds upon this class)

```
class Address extends AppModel {

     // explicitely define the name of model
     var $name = "Address";

}
```

in http://domain.tld/cake/, there is an .htaccess file that performs a mod-rewrite, which takes addresses and sends it to the address controller

CakePHP includes the basic framework (as long as you adhere to the conventions, it will do this all automatically)

To change what you see, must change controller, model, and view

CONTROLLER
remove scaffolding
put in a function index()
- this is required when scaffolding does not do it itself
- Controller puts all the data from the address model into an array addresses
(see code for cake4)
- also a function view($id) with an argument id to print out the info
    - domain.tld/cake4/addresses/view/7
    - this means that it accesses the addresses controller, with function view, and 7 as the argument passed in to function view
- function for edit($id = null) as well
    - setting $id to null is just a default

you can put methods on a controller by declaring a function (like edit, add, delete, etc.)

VIEWS
index.thtml - just a templated html file with php to echo out all the parts of the array we populated

flash method shows text as a link

cakePHP has an html helper class (also has ajax, email sending, etc.)
- classes that help you get stuff done
input is a method of html helper class that helps you store data
Check the API for more details

for validation, define an associative array to compare against

```
var $validate = array(
     'first_name' => VALID_NOT_EMPTY
     ..
     )
```

if you download cakePHP and just make the basic scaffold, you can use bake, and it will output the code (from the scaffold) automatically.
php -f cake/scripts/bake.php (from your public_html)

you can do similar things with jquery.
download the jquery library, link to it, and you're ready to roll

an example of using jquery

```
funcion disableAllButtons() {
     $(":submit").attr("disabled","disabled");
}
```

jquery uses the dollar sign as a function name. This finds the submit buttons, and sets the attribute disabled to disabled.
- also supports x-path-like selectors

```
$("input[@name!='submit1']")...
```

To do ajax in jquery,

```
function doAjax(id) {
     $.getJSON("json.php?id=" + id,
          function(data) { $('#bar').html(data.servertime); } );
}
```