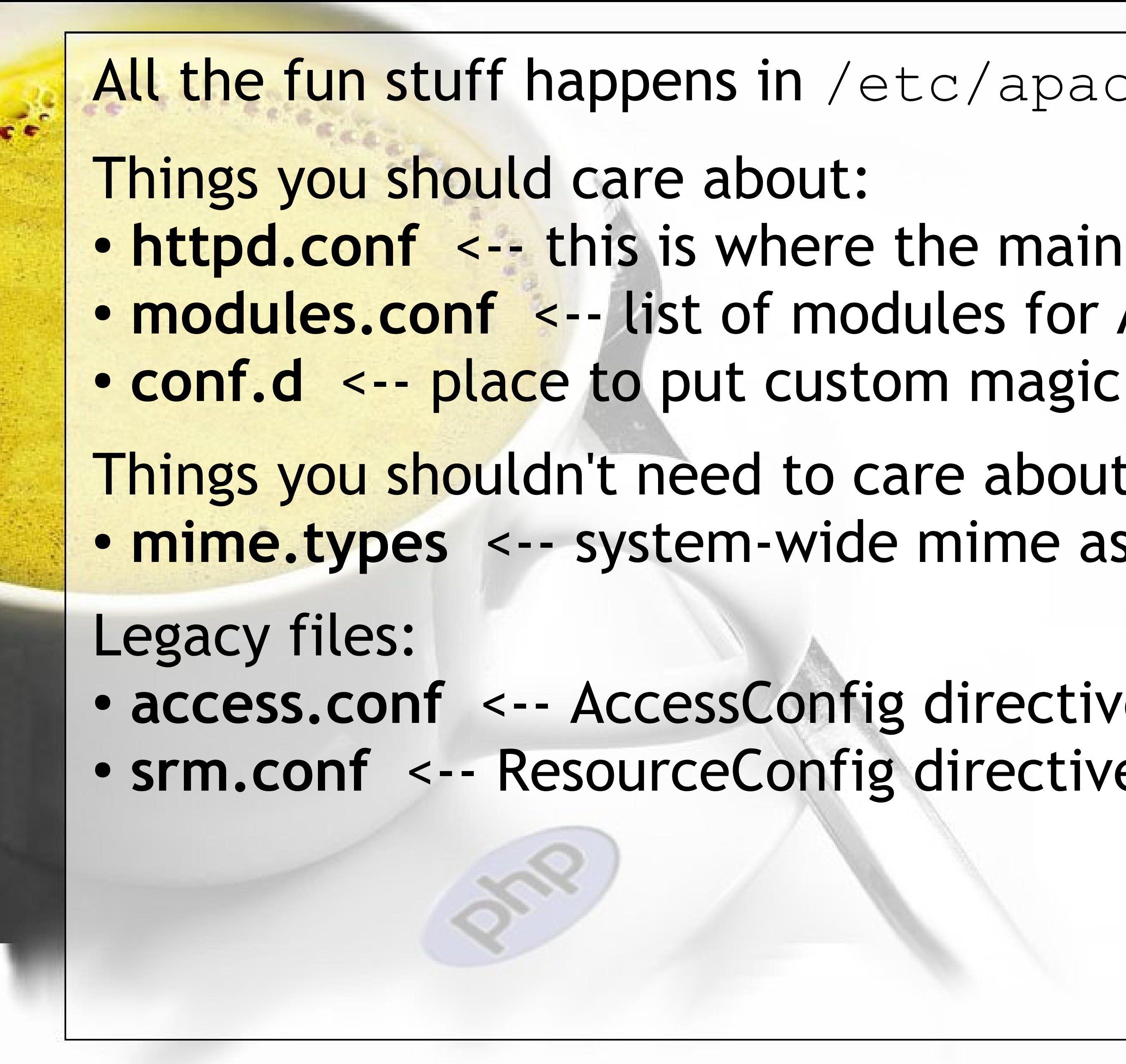
Apache Configuration Primer

Jonathan Oxer

December 7th, 2006
Open Source Developers Conference
Melbourne, Australia

Quick Intro: Apache 1.3 Configuration

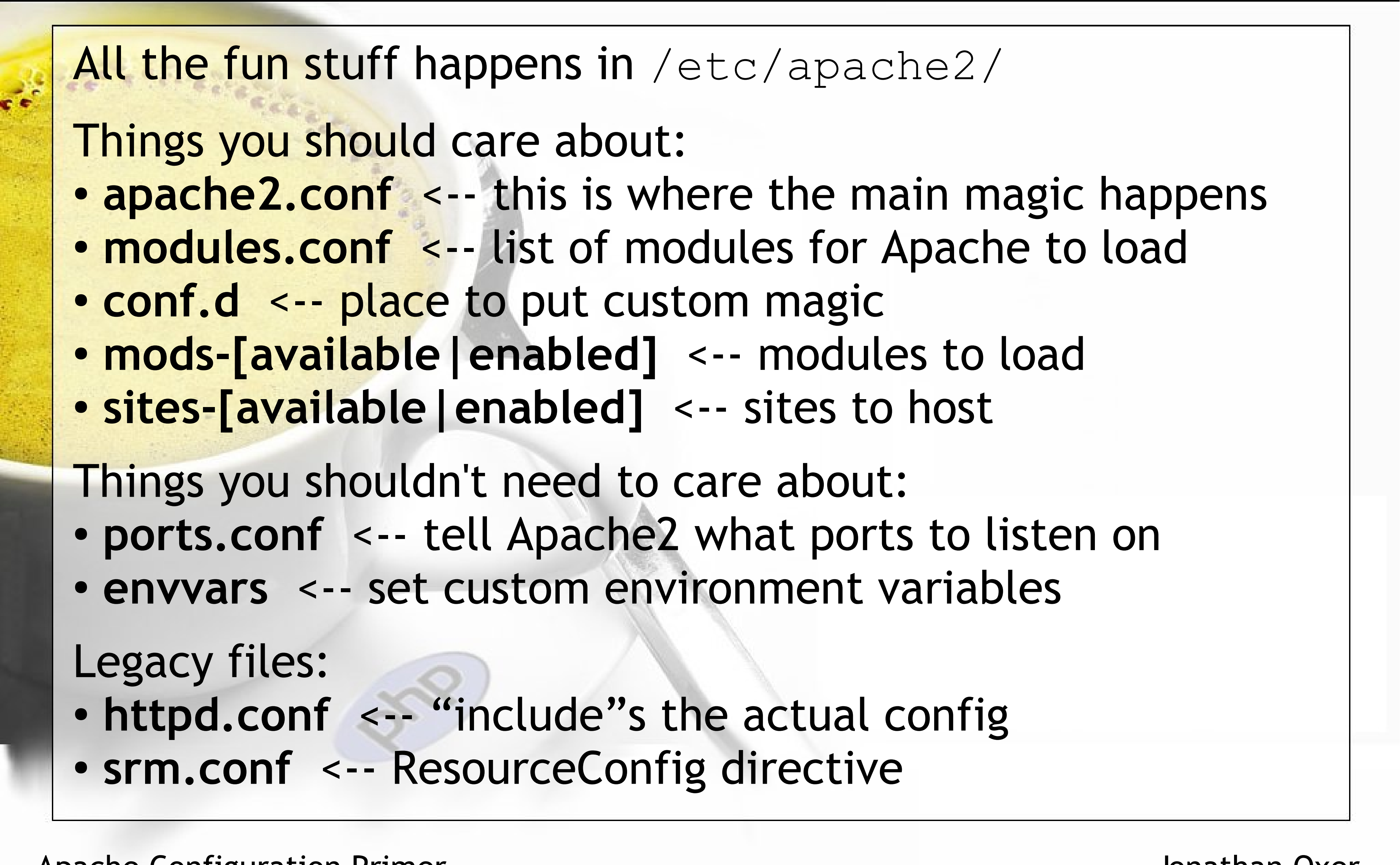


All the fun stuff happens in `/etc/apache/`

Things you should care about:
- **httpd.conf** <-- this is where the main magic happens
- **modules.conf** <-- list of modules for Apache to load
- **conf.d** <-- place to put custom magic

Things you shouldn't need to care about:
- **mime.types** <-- system-wide mime associations

Legacy files:
- **access.conf** <-- AccessConfig directive
- **srm.conf** <-- ResourceConfig directive



Quick Intro: Apache 2.0 Configuration

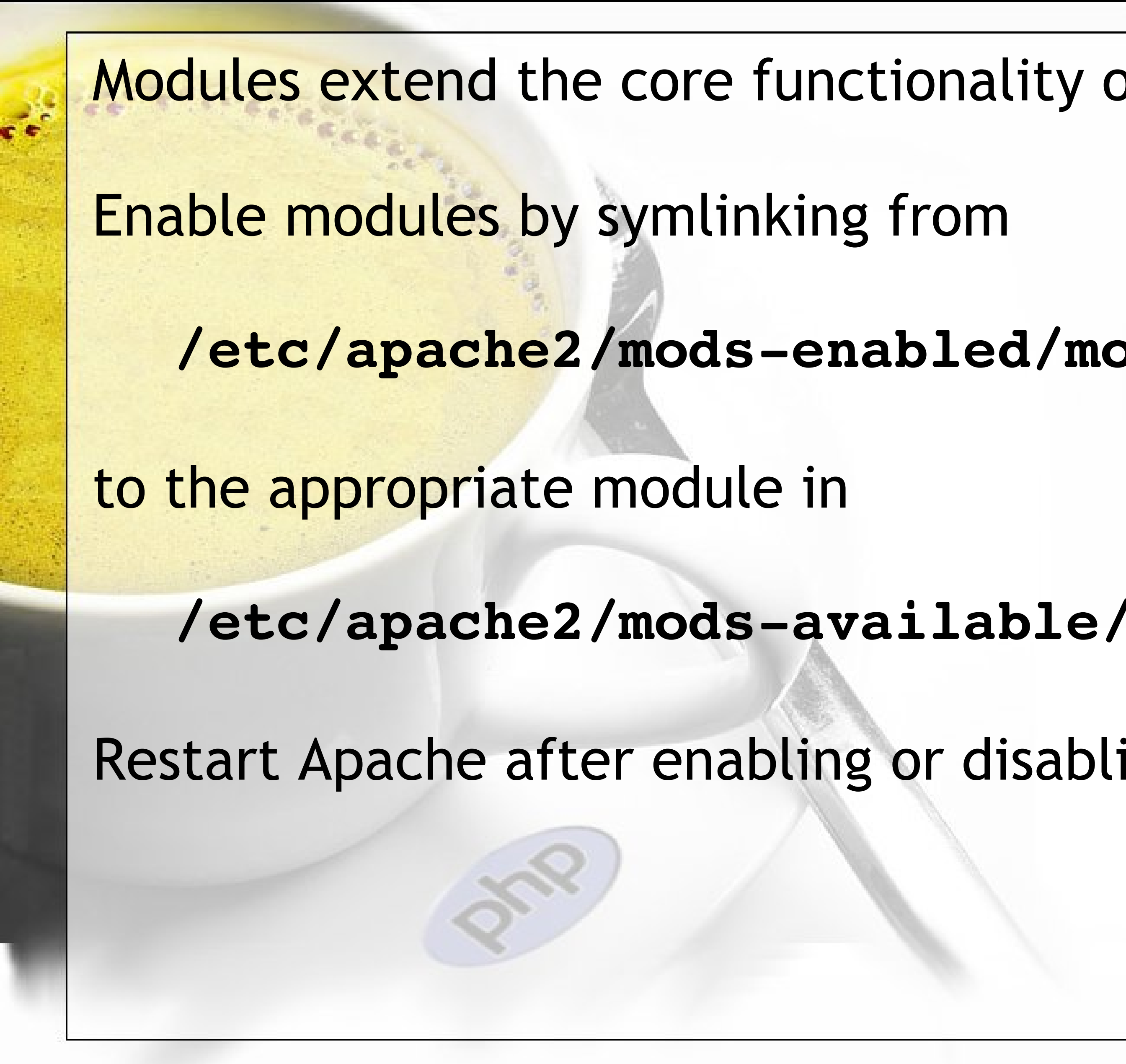
All the fun stuff happens in `/etc/apache2/`

Things you should care about:
- **apache2.conf** <-- this is where the main magic happens
- **modules.conf** <-- list of modules for Apache to load
- **conf.d** <-- place to put custom magic
- **mods-[available|enabled]** <-- modules to load
- **sites-[available|enabled]** <-- sites to host

Things you shouldn't need to care about:
- **ports.conf** <-- tell Apache2 what ports to listen on
- **envvars** <-- set custom environment variables

Legacy files:
- **httpd.conf** <-- “include”s the actual config
- **srm.conf** <-- ResourceConfig directive

Apache Modules



Modules extend the core functionality of Apache.

Enable modules by symlinking from

```
   /etc/apache2/mods-enabled/modulename
```

to the appropriate module in

```
   /etc/apache2/mods-available/<module>
```

Restart Apache after enabling or disabling modules.

Apache Modules

Or use the helper scripts:

```
a2enmod <module name>
a2dismod <module name>
```

Restart Apache after enabling or disabling modules.

```
/etc/init.d/apache2 restart
```

or

```
apache2ctl restart
```

Traditional Name-Based Virtual Hosts

Require a configuration block for every virtual host.

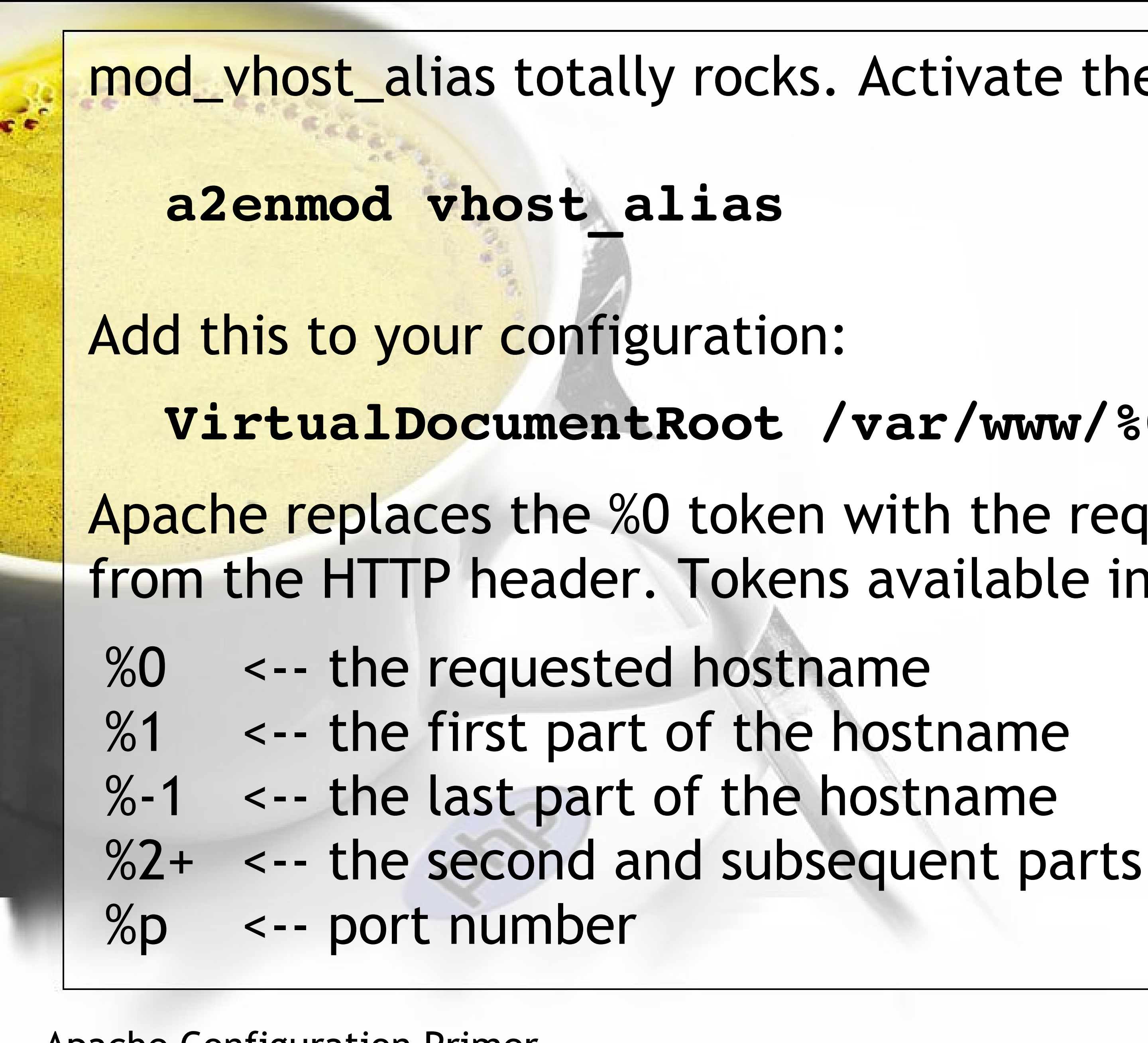
Store them in

```
/etc/apache2/sites-available/<sitename>
```

Enable or disable with

```
a2ensite <sitename>
a2dissite <sitename>
```

Dynamic Virtual Hosts



mod_vhost_alias totally rocks. Activate the module:

```
   a2enmod vhost_alias
```

Add this to your configuration:

```
   VirtualDocumentRoot /var/www/%0
```

Apache replaces the %0 token with the requested hostname from the HTTP header. Tokens available include:

%0	<-- the requested hostname
%1	<-- the first part of the hostname
%-1	<-- the last part of the hostname
%2+	<-- the second and subsequent parts of the hostname
%p	<-- port number

Dynamic Virtual Host Logging

Dynamic vhosts can't have separate logfiles.

Add a “virtual host” field to your logfile like this:

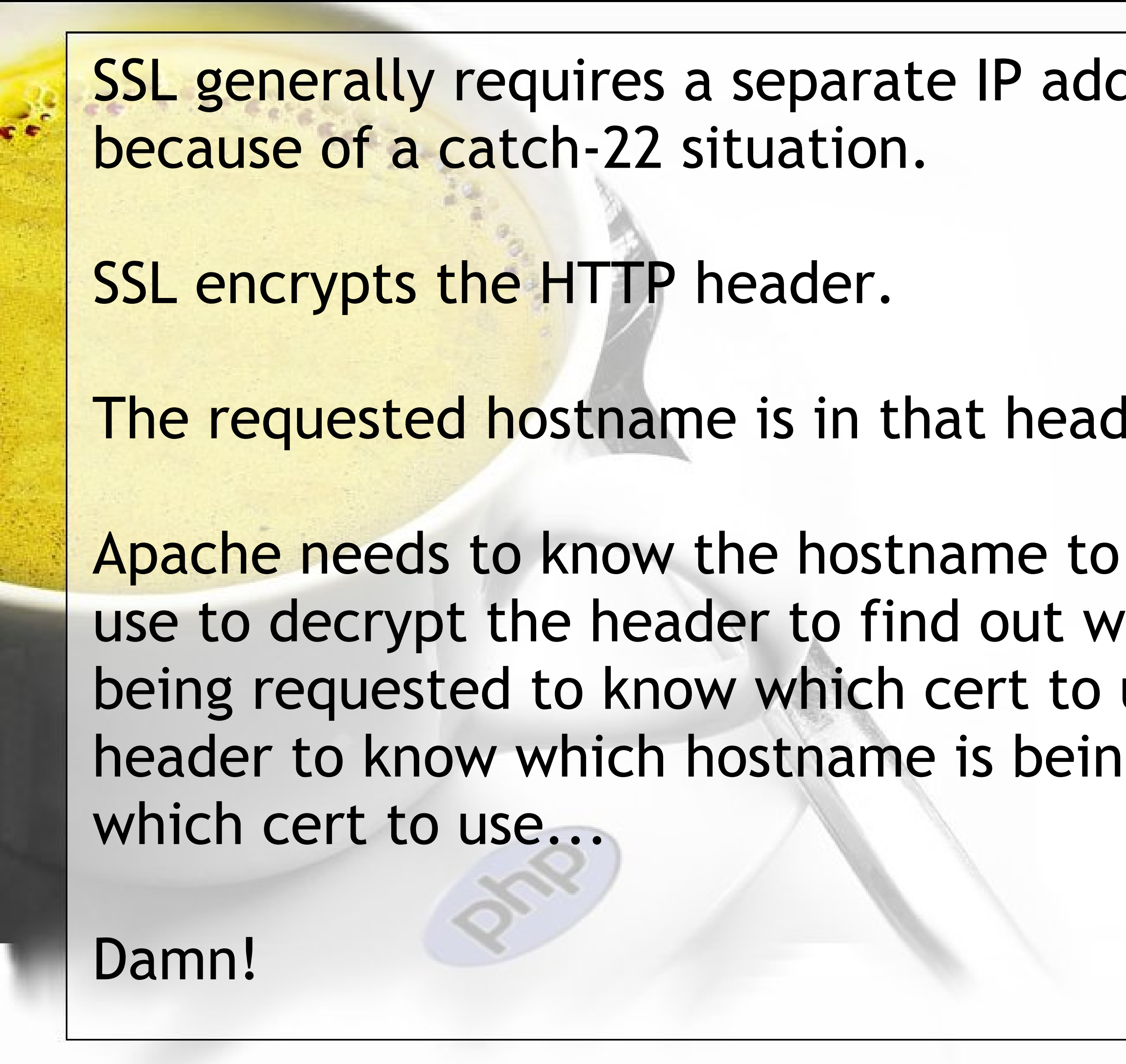
```
LogFormat "%V %h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" vcombined
CustomLog /var/log/apache2/access.log vcombined
```

Then you can use the `split-logfile` script to, well, split the logfile.

Finally, restart Apache:

```
/etc/init.d/apache2 restart
```

Traditional SSL: IP-based Vhosts



SSL generally requires a separate IP address per vhost because of a catch-22 situation.

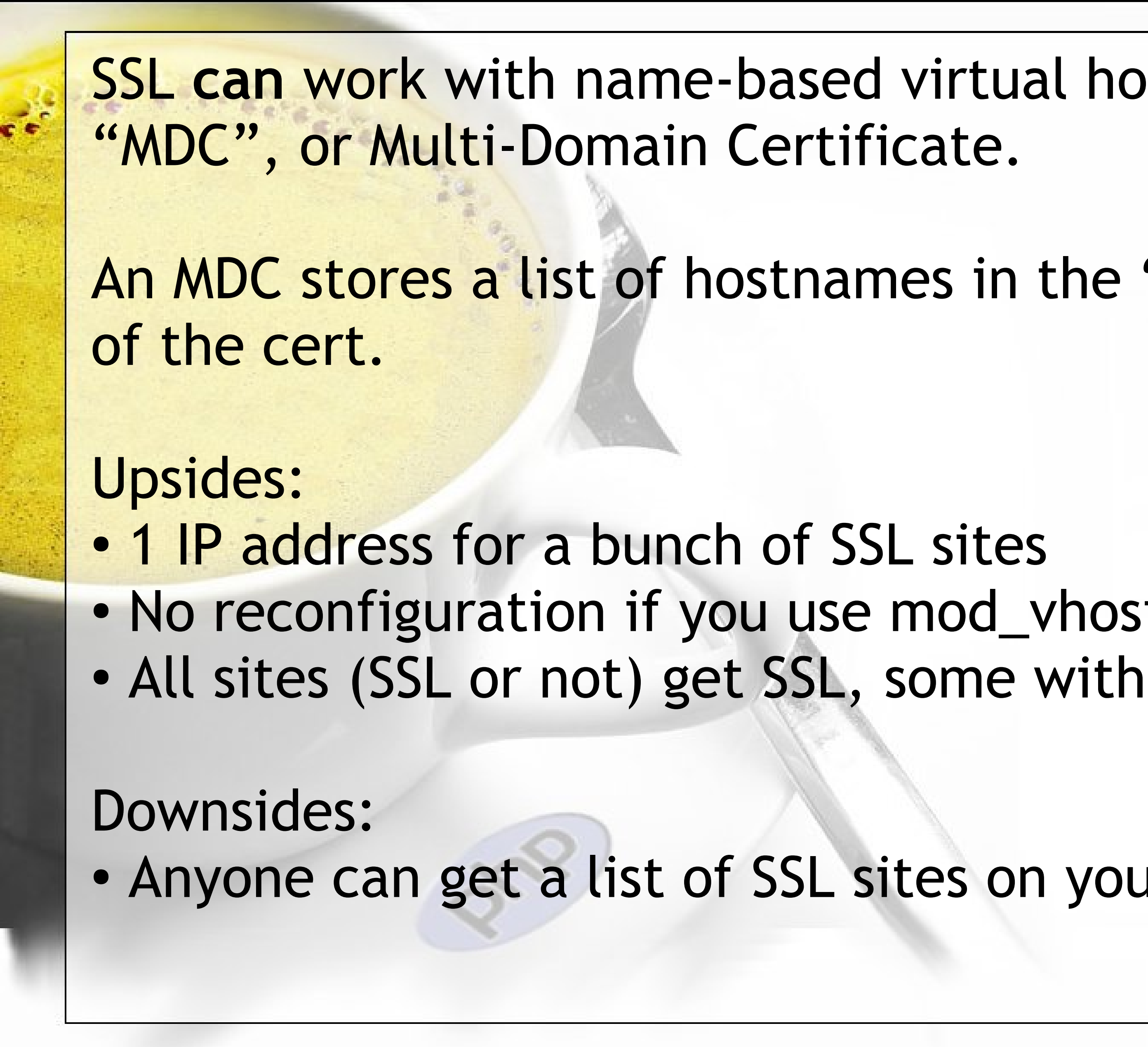
SSL encrypts the HTTP header.

The requested hostname is in that header.

Apache needs to know the hostname to know which cert to use to decrypt the header to find out which hostname is being requested to know which cert to use to decrypt the header to know which hostname is being requested to know which cert to use...

Damn!

SSL with Name-Based Vhosts



SSL **can** work with name-based virtual hosts if you use an "MDC", or Multi-Domain Certificate.

An MDC stores a list of hostnames in the "Subject" element of the cert.

Upsides:
- 1 IP address for a bunch of SSL sites
- No reconfiguration if you use mod_vhost_alias
- All sites (SSL or not) get SSL, some with a mismatched cert

Downsides:
- Anyone can get a list of SSL sites on your server

More Information

These slides are online at:
jon.oxer.com.au/talks

Dynamic vhost config directives:
httpd.apache.org/docs/2.0/mod/mod_vhost_alias.html