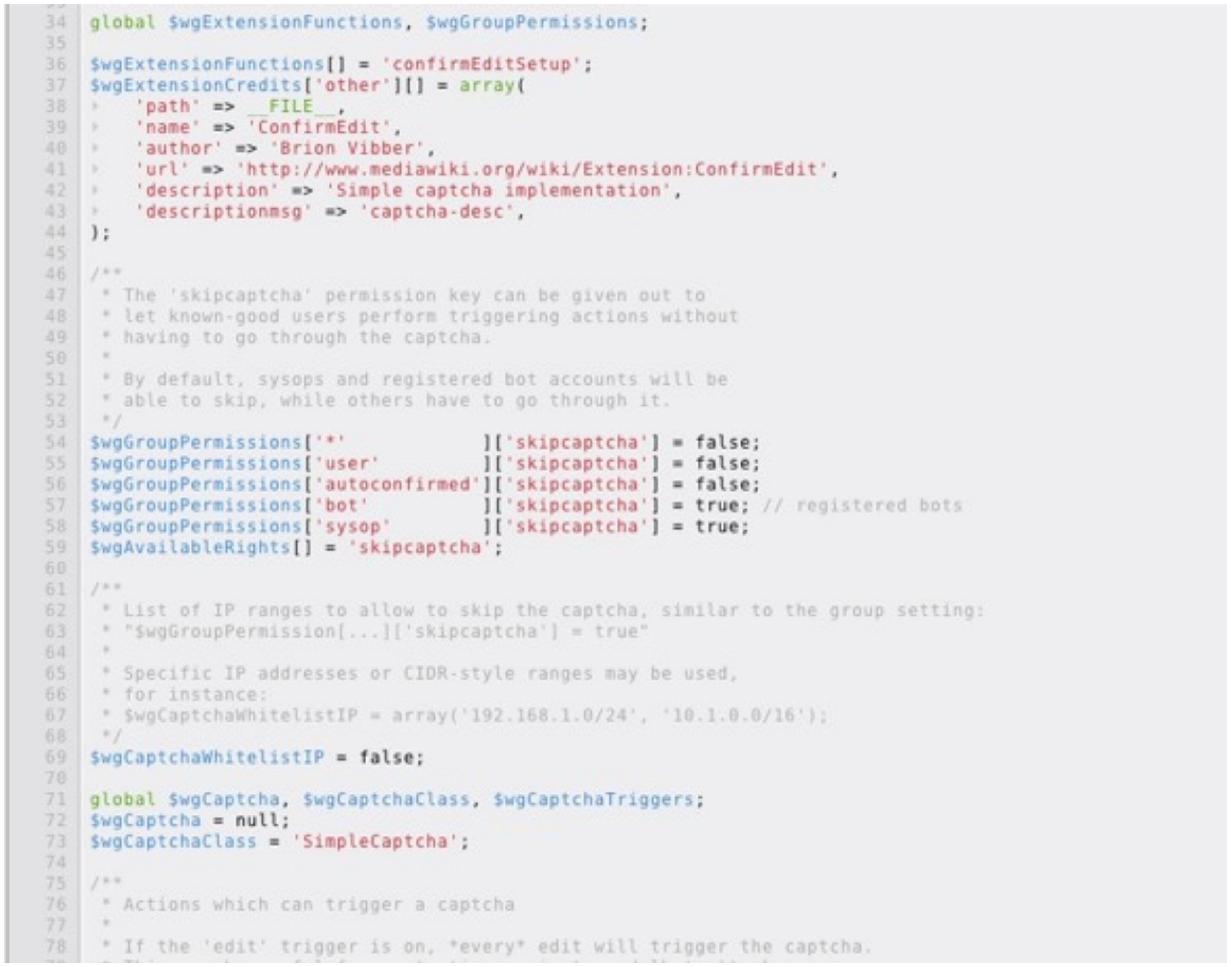


```php
global $wgExtensionFunctions, $wgGroupPermissions;

$wgExtensionFunctions[] = 'confirmEditSetup';
$wgExtensionCredits['other'][] = array(
	'path' => __FILE__,
	'name' => 'ConfirmEdit',
	'author' => 'Brion Vibber',
	'url' => 'http://www.mediawiki.org/wiki/Extension:ConfirmEdit',
	'description' => 'Simple captcha implementation',
	'descriptionmsg' => 'captcha-desc',
);

/**
 * The 'skipcaptcha' permission key can be given out to
 * let known-good users perform triggering actions without
 * having to go through the captcha.
 *
 * By default, sysops and registered bot accounts will be
 * able to skip, while others have to go through it.
 */
$wgGroupPermissions['*'            ]['skipcaptcha'] = false;
$wgGroupPermissions['user'         ]['skipcaptcha'] = false;
$wgGroupPermissions['autoconfirmed']['skipcaptcha'] = false;
$wgGroupPermissions['bot'          ]['skipcaptcha'] = true; // registered bots
$wgGroupPermissions['sysop'        ]['skipcaptcha'] = true;
$wgAvailableRights[] = 'skipcaptcha';

/**
 * List of IP ranges to allow to skip the captcha, similar to the group setting:
 * "$wgGroupPermission[...]['skipcaptcha'] = true"
 *
 * Specific IP addresses or CIDR-style ranges may be used,
 * for instance:
 * $wgCaptchaWhitelistIP = array('192.168.1.0/24', '10.1.0.0/16');
 */
$wgCaptchaWhitelistIP = false;

global $wgCaptcha, $wgCaptchaClass, $wgCaptchaTriggers;
$wgCaptcha = null;
$wgCaptchaClass = 'SimpleCaptcha';

/**
 * Actions which can trigger a captcha
 *
 * If the 'edit' trigger is on, *every* edit will trigger the captcha.
```

A brief introduction to MediaWiki extension development

Andrew Garrett
Wikimedia Foundation

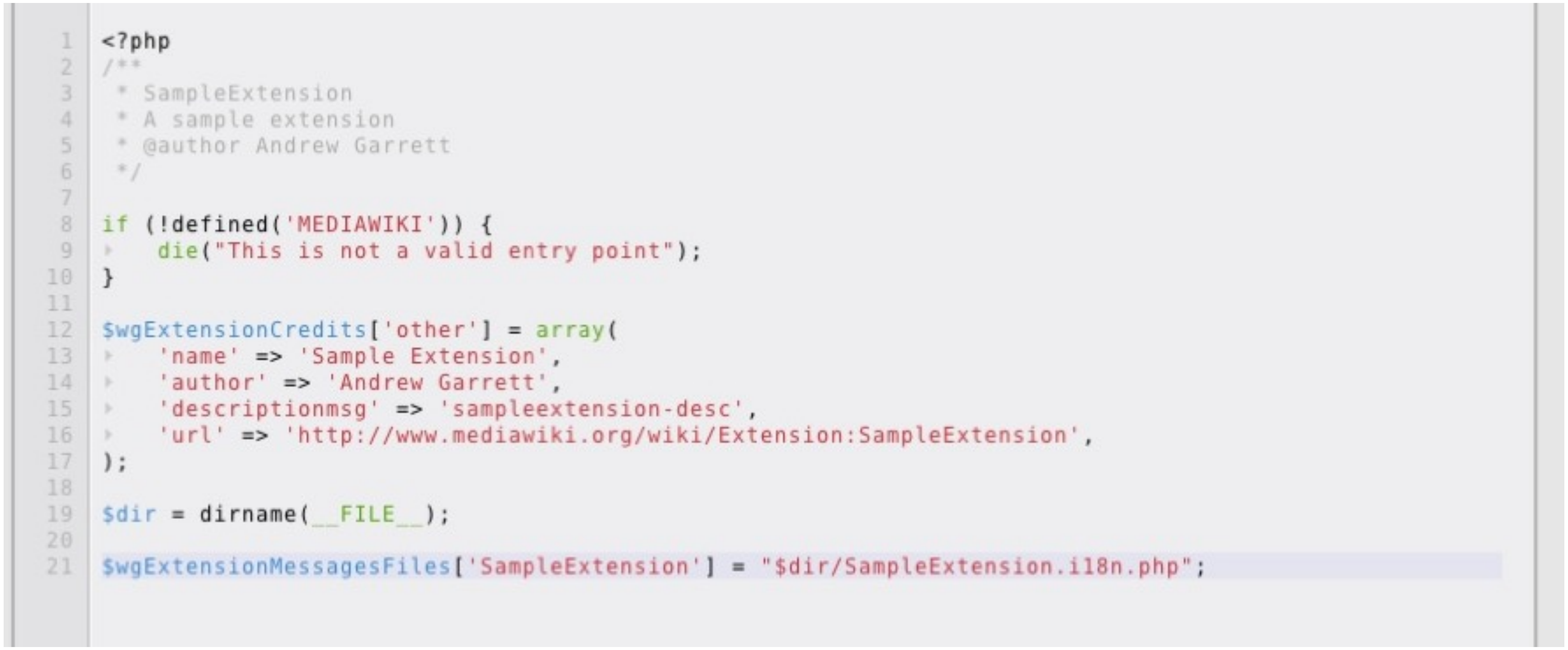
How does a MediaWiki extension fit together?

- Has a directory (name is upper camel case)
- Has a setup file (included from LocalSettings.php)
- Registers hooks, special pages, extension functions, localization files, etc with global variables
- Defines default configuration settings
- For bigger extensions, other code is in separate files

```
MyExtension/MyExtension.php

MyExtension/MyExtension.i18n.php

MyExtension/MyExtension_body.php

MyExtension/MyExtension.hooks.php

MyExtension/SpecialMyExtension.php
```

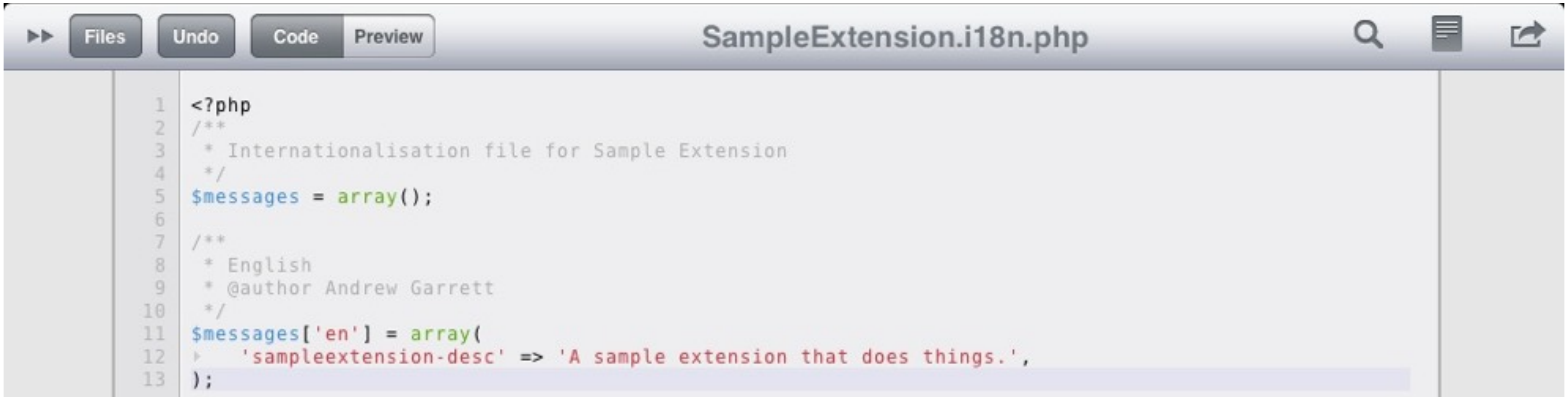
Where do you start?

- Create the extension directory
- Create skeleton setup and localization files
- Make it do something!


```php
<?php
/**
 * SampleExtension
 * A sample extension
 * @author Andrew Garrett
 */

if (!defined('MEDIAWIKI')) {
	die("This is not a valid entry point");
}

$wgExtensionCredits['other'] = array(
	'name' => 'Sample Extension',
	'author' => 'Andrew Garrett',
	'descriptionmsg' => 'sampleextension-desc',
	'url' => 'http://www.mediawiki.org/wiki/Extension:SampleExtension',
);

$dir = dirname(__FILE__);

$wgExtensionMessagesFiles['SampleExtension'] = "$dir/SampleExtension.i18n.php";
```

Internationalisation

- Anything that is shown to the user (except URLs) should be localisable.
- At Wikimedia, we use messages for localisation.
- If you use messages for localisation, they will be translated at translatewiki.

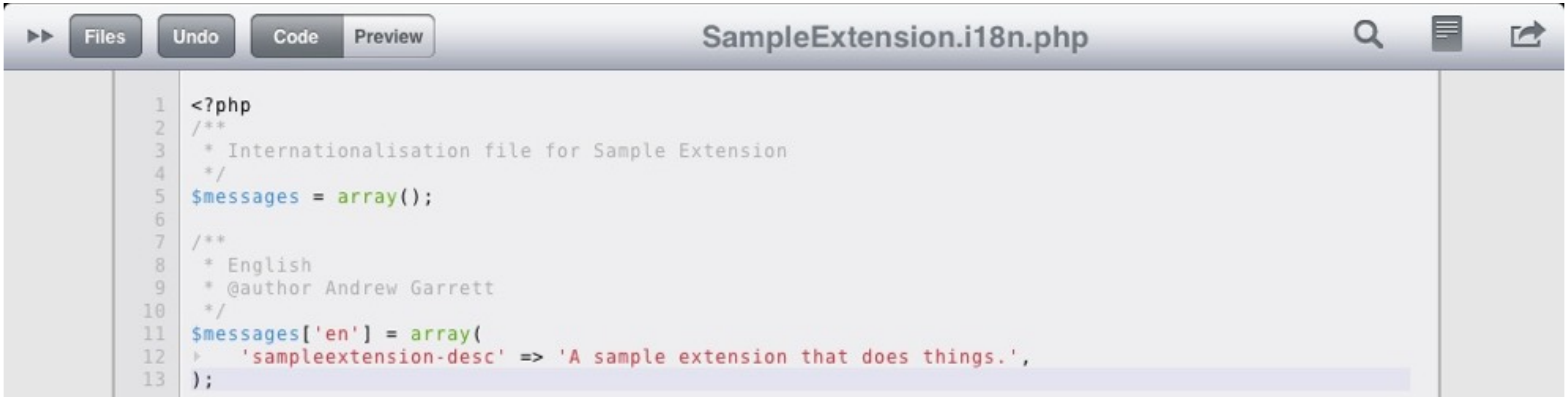


```php
<?php
/**
 * Internationalisation file for Sample Extension
 */
$messages = array();

/**
 * English
 * @author Andrew Garrett
 */
$messages['en'] = array(
	'sampleextension-desc' => 'A sample extension that does things.',
);
```

Ten minutes of MediaWiki coding

- Internationalisation
- Input and output (GET/POST variables, outputting HTML)
- Security
- Resources
  - Class documentation at http://svn.wikimedia.org/doc
  - Other documentation at http://www.mediawiki.org

Step 1: Add your message



```php
<?php
/**
 * Internationalisation file for Sample Extension
 */
$messages = array();

/**
 * English
 * @author Andrew Garrett
 */
$messages['en'] = array(
	'sampleextension-desc' => 'A sample extension that does things.',
);
```

Step 2: Use your message

- To use a message, use `wfMsg('message-name')`
- To get a message, and parse wikitext to HTML, use `wfMessage('message-name')->parsed()`
- You can insert arguments by passing an array as the second parameter to `wfMessage` or by adding extra parameters to `wfMsg`
- `wfMessage( 'message-name', array( 1, 2 ) )`
- `wfMsg( 'message-name', 1, 2 )`
- `wfMessage` returns a `Message` object

Input and Output

- In MediaWiki, an `OutputPage` object represents the output from a page view operation.
  - In most cases, use `$wgOut`
  - Add HTML to the output with `$wgOut->addHTML()`
  - OutputPage also has methods to add JS and CSS, resource modules, HTTP headers, redirect and set caching policy.

Input and Output

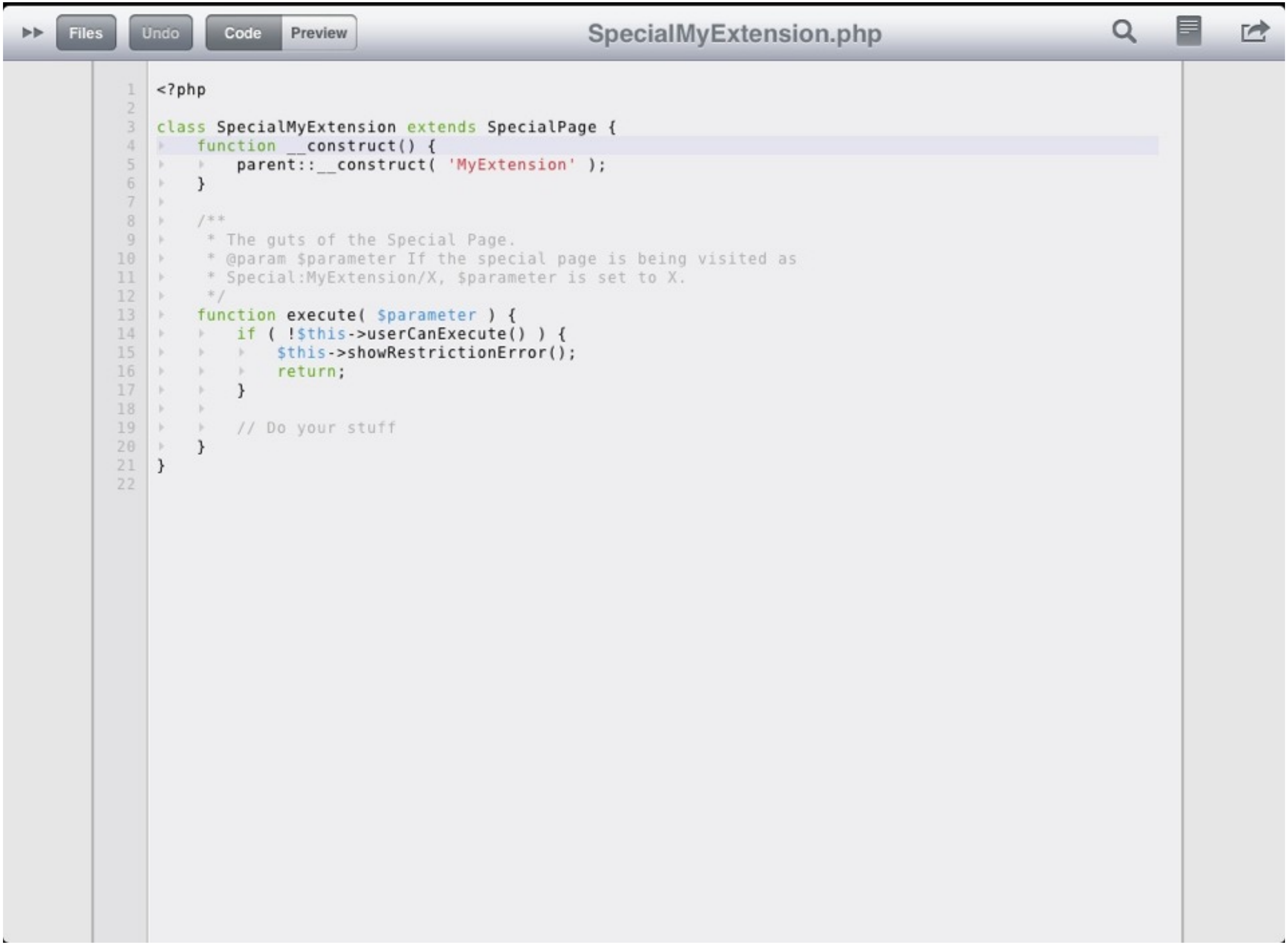
- In MediaWiki. `$wgRequest` is a WebRequest object which wraps `$_GET`/`$_POST`/ `$_COOKIE`
  - Use `getVal()`, `getCheck()`, `getInt()`, etc
  - You can tell if a request was posted with `wasPosted()`

Security

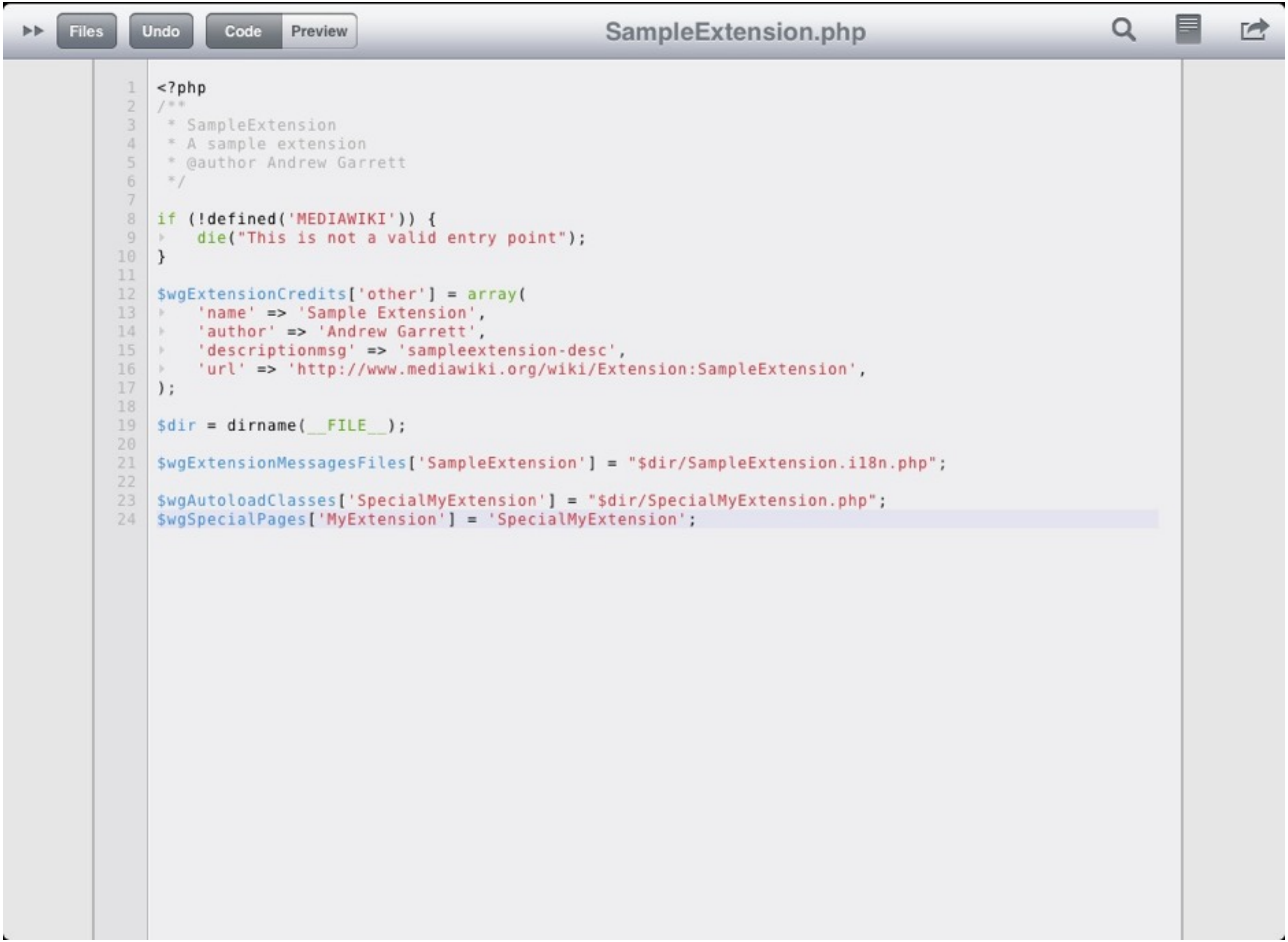
- Read http://www.mediawiki.org/wiki/Security_for_developers
- Basics:
  - Cross site scripting
  - Database wrappers

What can an extension do?

- Create special pages
- Extend wiki markup
- Hook events



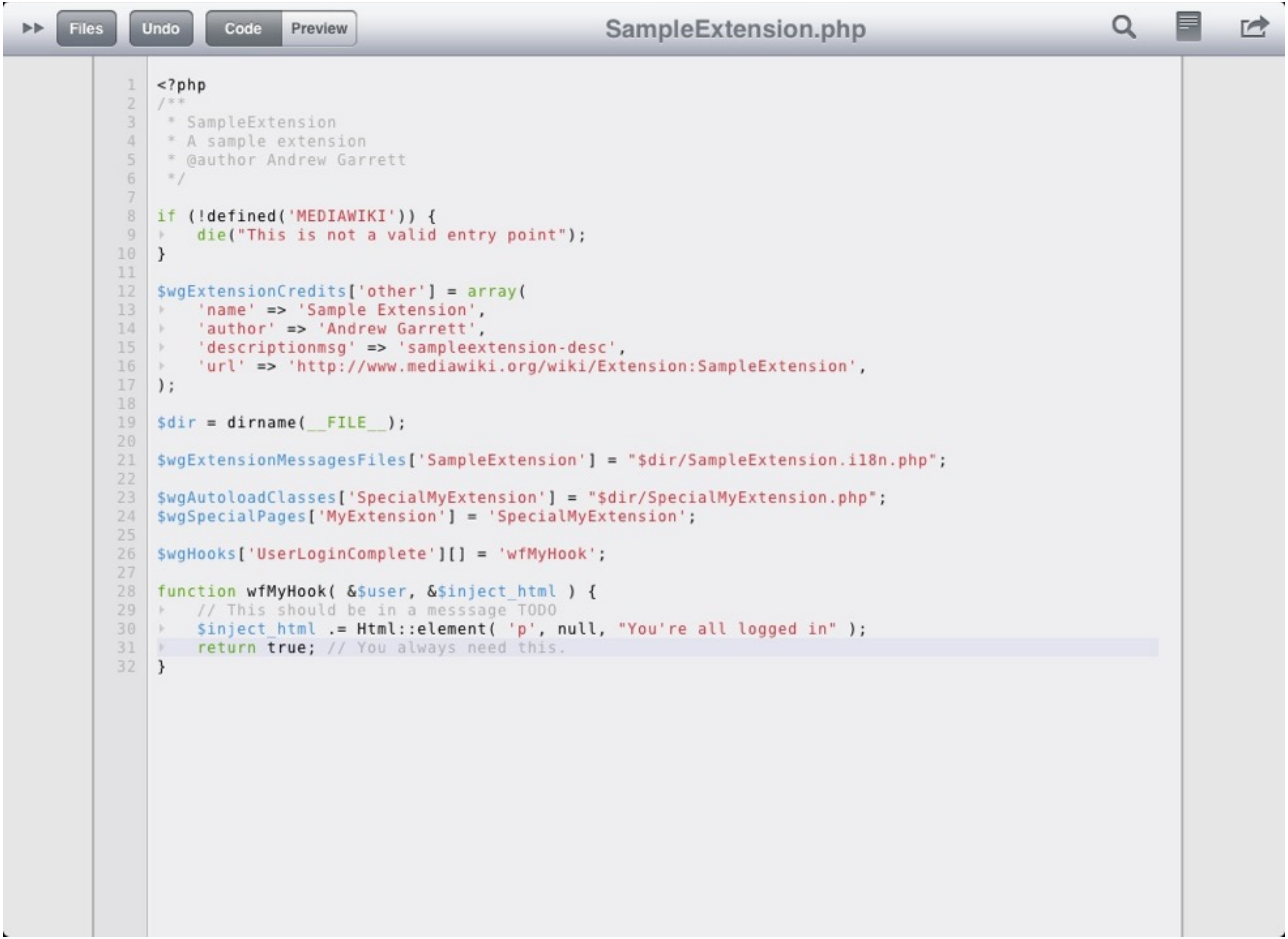
```php
<?php

class SpecialMyExtension extends SpecialPage {
	function __construct() {
		parent::__construct( 'MyExtension' );
	}

	/**
	 * The guts of the Special Page.
	 * @param $parameter If the special page is being visited as
	 * Special:MyExtension/X, $parameter is set to X.
	 */
	function execute( $parameter ) {
		if ( !$this->userCanExecute() ) {
			$this->showRestrictionError();
			return;
		}

		// Do your stuff
	}
}

```



```php
<?php
/**
 * SampleExtension
 * A sample extension
 * @author Andrew Garrett
 */

if (!defined('MEDIAWIKI')) {
	die("This is not a valid entry point");
}

$wgExtensionCredits['other'] = array(
	'name' => 'Sample Extension',
	'author' => 'Andrew Garrett',
	'descriptionmsg' => 'sampleextension-desc',
	'url' => 'http://www.mediawiki.org/wiki/Extension:SampleExtension',
);

$dir = dirname(__FILE__);

$wgExtensionMessagesFiles['SampleExtension'] = "$dir/SampleExtension.i18n.php";

$wgAutoloadClasses['SpecialMyExtension'] = "$dir/SpecialMyExtension.php";
$wgSpecialPages['MyExtension'] = 'SpecialMyExtension';
```

What is a hook?

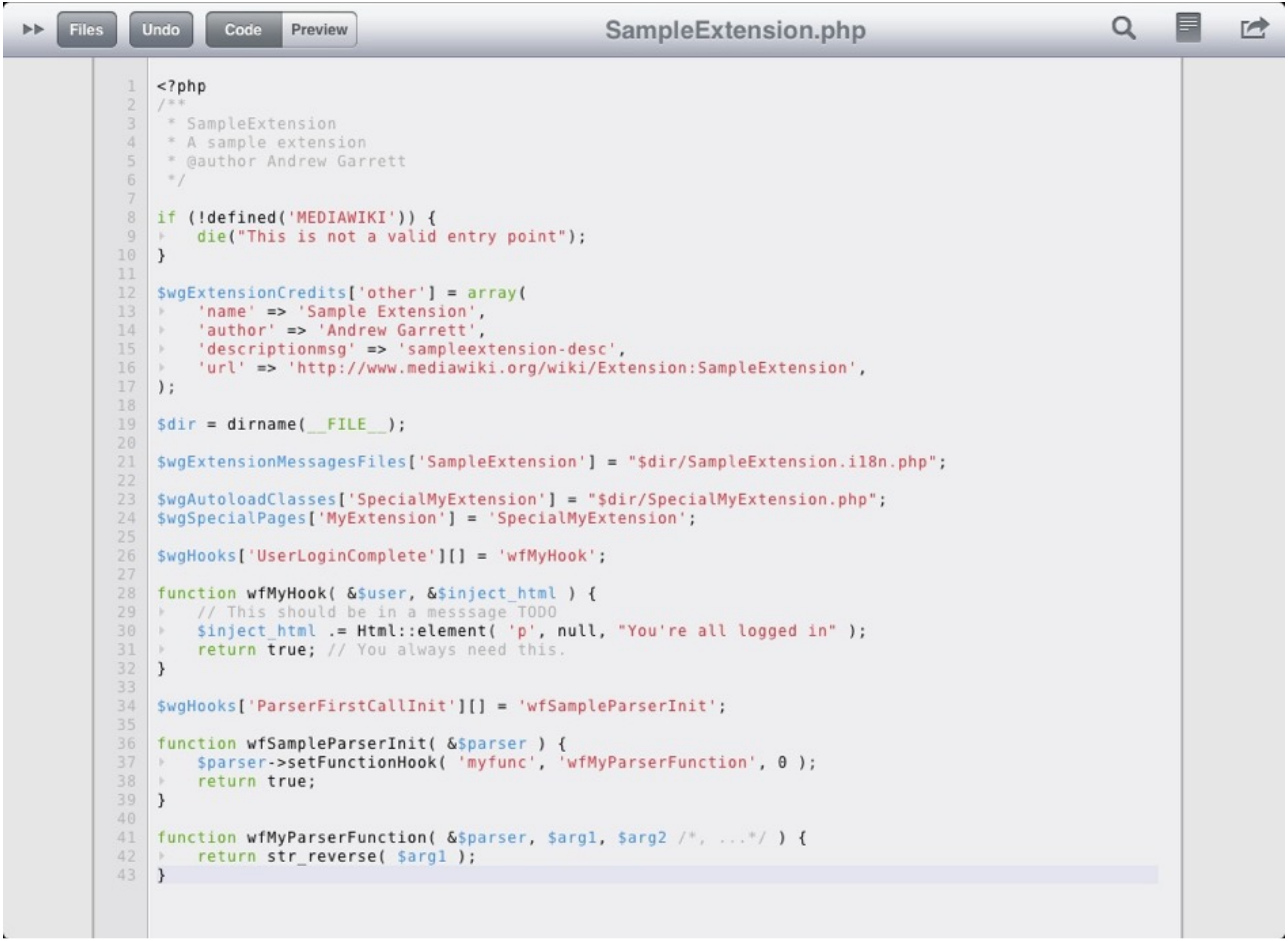
- Allows you to intercept MediaWiki events and do your own thing.
- List of available hooks at http://www.mediawiki.org/wiki/Manual:Hooks
- Create a handler using the template on the hook's page, and register it in `$wgHooks`.



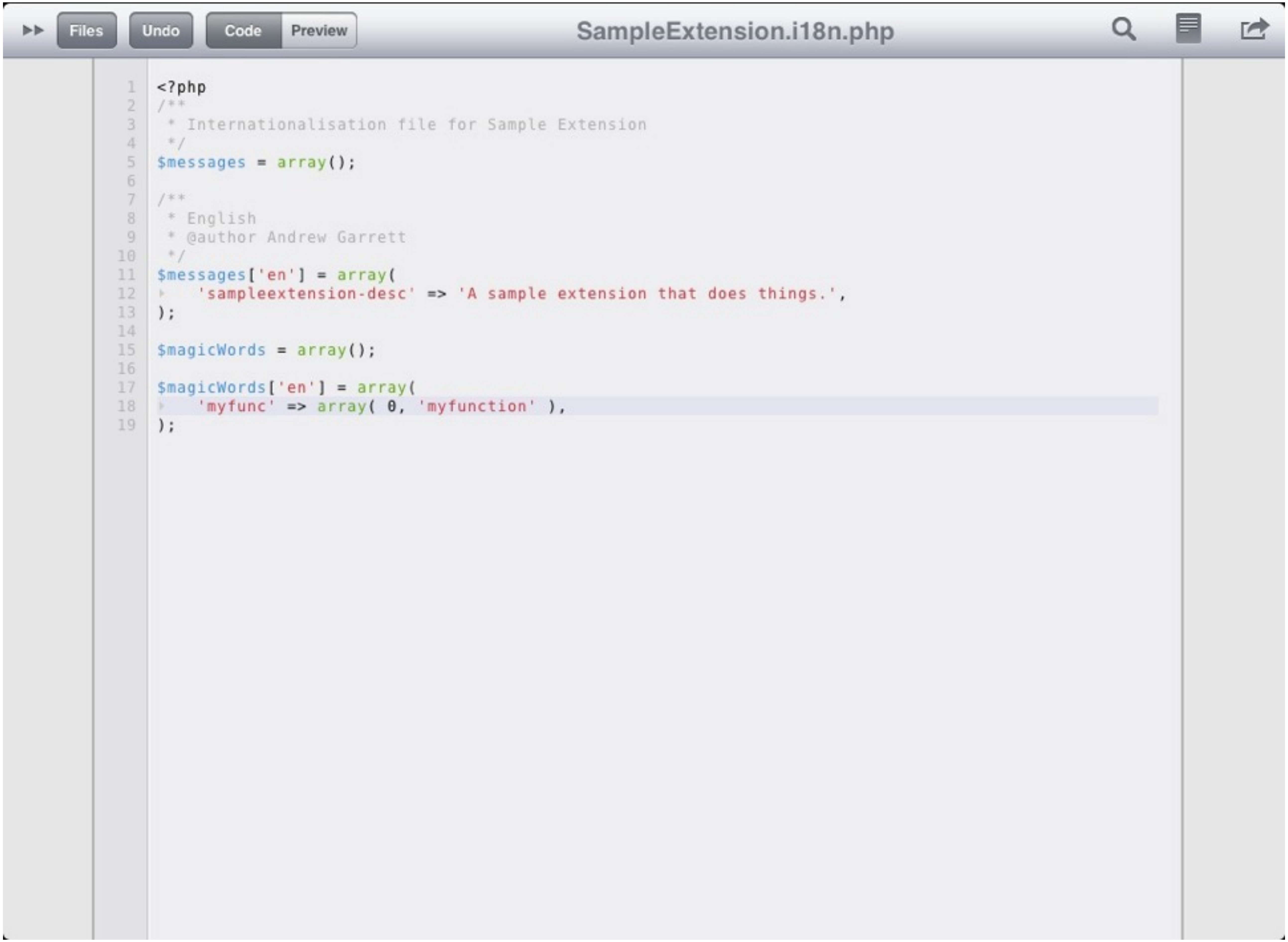
```php
<?php
/**
 * SampleExtension
 * A sample extension
 * @author Andrew Garrett
 */

if (!defined('MEDIAWIKI')) {
	die("This is not a valid entry point");
}

$wgExtensionCredits['other'] = array(
	'name' => 'Sample Extension',
	'author' => 'Andrew Garrett',
	'descriptionmsg' => 'sampleextension-desc',
	'url' => 'http://www.mediawiki.org/wiki/Extension:SampleExtension',
);

$dir = dirname(__FILE__);

$wgExtensionMessagesFiles['SampleExtension'] = "$dir/SampleExtension.i18n.php";

$wgAutoloadClasses['SpecialMyExtension'] = "$dir/SpecialMyExtension.php";
$wgSpecialPages['MyExtension'] = 'SpecialMyExtension';

$wgHooks['UserLoginComplete'][] = 'wfMyHook';

function wfMyHook( &$user, &$inject_html ) {
	// This should be in a messsage TODO
	$inject_html .= Html::element( 'p', null, "You're all logged in" );
	return true; // You always need this.
}
```

Extending wiki markup

- Use the ParserFirstCallInit hook
- Call `$parser->setFunctionHook()` or `$parser->setHook()`
- Set up an appropriate callback, using `$parser` for context.
- Types of parser extensions:
    - Tags (return HTML)
    - Parser functions (return wikitext or HTML)



```php
<?php
/**
 * SampleExtension
 * A sample extension
 * @author Andrew Garrett
 */

if (!defined('MEDIAWIKI')) {
	die("This is not a valid entry point");
}

$wgExtensionCredits['other'] = array(
	'name' => 'Sample Extension',
	'author' => 'Andrew Garrett',
	'descriptionmsg' => 'sampleextension-desc',
	'url' => 'http://www.mediawiki.org/wiki/Extension:SampleExtension',
);

$dir = dirname(__FILE__);

$wgExtensionMessagesFiles['SampleExtension'] = "$dir/SampleExtension.i18n.php";

$wgAutoloadClasses['SpecialMyExtension'] = "$dir/SpecialMyExtension.php";
$wgSpecialPages['MyExtension'] = 'SpecialMyExtension';

$wgHooks['UserLoginComplete'][] = 'wfMyHook';

function wfMyHook( &$user, &$inject_html ) {
	// This should be in a messsage TODO
	$inject_html .= Html::element( 'p', null, "You're all logged in" );
	return true; // You always need this.
}

$wgHooks['ParserFirstCallInit'][] = 'wfSampleParserInit';

function wfSampleParserInit( &$parser ) {
	$parser->setFunctionHook( 'myfunc', 'wfMyParserFunction', 0 );
	return true;
}

function wfMyParserFunction( &$parser, $arg1, $arg2 /*, ...*/ ) {
	return str_reverse( $arg1 );
}
```



```php
<?php
/**
 * Internationalisation file for Sample Extension
 */
$messages = array();

/**
 * English
 * @author Andrew Garrett
 */
$messages['en'] = array(
	'sampleextension-desc' => 'A sample extension that does things.',
);

$magicWords = array();

$magicWords['en'] = array(
	'myfunc' => array( 0, 'myfunction' ),
);
```

Question and Hacking Time

Tell me what you want to know about