Build Status

Collection of common Sinatra extensions, semi-officially supported.

Goals

- For every future Sinatra release, have at least one fully compatible release
- High code quality, high test coverage
- Include plugins people usually ask for a lot

Included extensions

Common Extensions

These are common extension which will not add significant overhead or change any behavior of already existing APIs. They do not add any dependencies not already installed with this gem.

Currently included:

- sinatra/capture: Let's you capture the content of blocks in templates.
- sinatra/config_file: Allows loading configuration from yaml files.
- sinatra/content_for: Adds Rails-style content_for helpers to Haml, Erb, Erubis and Slim.
- sinatra/cookies: A cookies helper for reading and writing cookies.
- sinatra/engine_tracking: Adds methods like haml? that allow helper methods to check whether they are called from within a template.
- sinatra/json: Adds a #json helper method to return JSON documents.
- sinatra/link_header: Helpers for generating link HTML tags and corresponding Link HTTP headers. Adds link, stylesheet and prefetch helper methods.
- sinatra/multi_route: Adds ability to define one route block for multiple routes and multiple or custom HTTP verbs.
- sinatra/namespace: Adds namespace support to Sinatra.
- sinatra/respond_with: Choose action and/or template automatically depending on the incoming request. Adds helpers respond_to and respond_with.
- sinatra/custom_logger: This extension allows you to define your own logger instance using +logger+ setting. That logger then will be available as #logger helper method in your routes and views.

Custom Extensions

These extensions may add additional dependencies and enhance the behavior of the existing APIs.

Currently included:

- sinatra/decompile: Recreates path patterns from Sinatra's internal data structures (used by other extensions).
- sinatra/reloader: Automatically reloads Ruby files on code changes.

Other Tools

- sinatra/extension: Mixin for writing your own Sinatra extensions.
- sinatra/test_helpers: Helper methods to ease testing your Sinatra application. Partly extracted from Sinatra. Testing framework agnostic

Installation

Add gem 'sinatra-contrib' to *Gemfile*, then execute bundle install.

If you don't use Bundler, install the gem manually by executing gem install sinatra-contrib in your command line.

Usage

Classic Style

A single extension (example: sinatra-content-for):

```
require 'sinatra'
require 'sinatra/content_for'
```

Common extensions:

```
require 'sinatra'
require 'sinatra/contrib'
```

All extensions:

```
require 'sinatra'
require 'sinatra/contrib/all'
```

Modular Style

A single extension (example: sinatra-content-for):

```
require 'sinatra/base'
```

```
require 'sinatra/content_for'
require 'sinatra/namespace'

class MyApp < Sinatra::Base
  # Note: Some modules are extensions, some helpers, see the specific
  # documentation or the source
  helpers Sinatra::ContentFor
  register Sinatra::Namespace
end
```

Common extensions:

```
require 'sinatra/base'
require 'sinatra/contrib'

class MyApp < Sinatra::Base
  register Sinatra::Contrib
end
```

All extensions:

```
require 'sinatra/base'
require 'sinatra/contrib/all'

class MyApp < Sinatra::Base
  register Sinatra::Contrib
end
```

Documentation

For more info check the official docs and api docs.