Felix Felicis Documentation

Release 1.8

Hsiaoming Yang

February 09, 2015

Contents

Felix Felicis (aka liquidluck) is a static website generator. The name of "Felix Felicis" comes from Harry Porter; It is a magic potion that gives the drinker luck.

If you want to contribute to this project, fork liquidluck and send a pull request for your changes. Have a glance at *Development* section for details. If you just want to use it, you should watch the project for updates.

CHAPTER 1

User's Guide

1.1 Installation

If you are familiar with Python, it is strongly suggested that you install everything in virtualenv.

If you are a pythoner, and you have no idea about virtualenv, please do search the internet.

1.1.1 Distribute and Pip

If you are on Linux or Mac OS X, you are the lucky one:

```
$ sudo pip -U install liquidluck
```

If no pip available, try easy_install:

```
$ sudo easy_install -U liquidluck
```

Sorry, I have no knowledge about Windows, but it really works on Windows. Cygwin and MinGW would make a better life with UNIX software on Windows.

1.1.2 Install with GIT

If you prefer git, that is great. You can get the very latest code at GitHub:

```
$ git clone http://github.com/lepture/liquidluck.git
```

1.1.3 Mac User Attention

We use misaka (python wrapper for sundown) as the Markdown engine. It requires C compiler, which means you should install Xcode.

Then open Xcode's preference (command + ,), select *Downloads* tab, and install *Command Line Tools*.

I strongly suggest that you install the *Command Line Tools*, even if you don't use Felix Felicis. You will need it somewhere else.

1.1.4 Install on Windows

It is strongly suggested that you use Cygwin as the environment.

You should install these packages:

- python interpreters 2.x (or 3.x)
- make (Devel – the GNU version of make)
- gcc (gcc-core and gcc-g++)
- git (Devel – Fast version control ....)
- ca-certificates

Then head over to setuptools and install it, you will get `easy_install`.

1.2 Get Started

This section assumes that you already have Felix Felicis (liquidluck) installed. If you do not, header over to the *Installation* section.

1.2.1 Create a website

Now, you have Felix Felicis installed. Let's create a website:

```
$ cd ~
$ mkdir website
$ cd website
$ liquidluck init
```

You will be asked some questions. If the question has a default value, we don't change it, just hit Enter.

See what happens:

```
$ ls
content     settings.yml
```

1.2.2 Create a Post

Create a post:

```
$ touch content/hello-world.md
```

Write with you favorite editor, for example vim:

```
$ vim content/hello-world.md

# Hello World

- date: 2012-12-12
- category: life
- tags: python, code

----------------
```

```
Hello World. This is a DEMO post.
```

1.2.3 Build the website

Now that you have written a post, let's create the website:

```
$ cd ~/website
$ liquidluck build -v
$ ls
content     deploy      settings.py
```

The website is created, you can test your website:

```
$ liquidluck server
```

Open your browser: `http://127.0.0.1:8000`

Felix Felicis provided a more powerful *Preview Server*, you should check it.

1.2.4 Write more

You can test more posts yourself.

1.2.5 Learn Commands

Get all commands:

```
$ liquidluck -h
```

List all themes:

```
$ liquidluck search
```

Install a theme:

```
$ liquidlcuk install {{name}}
```

1.3 Writing

1.3.1 Markup

Felix Felicis (liquidluck) supports markup of Markdown and reStructuredText. It is suggested that you write in Markdown, it's easier.

There are three parts in each post:

- title – `Hello World` in the example
- meta – `date`, `category`, `tags` in the example
- content – everything below the first `-------`

An example in markdown:

~~~
# Felix Felicis                 <-------- this is title

- date: 2012-12-12 12:12        <-------- this is meta
- tags: python, blog, web

Here is the description.

--------------------------      <-------- this seprate meta and content

Hello World! Welcome to the     <-------- this is content
Felix Felicis World.

```                             <-------- this is normal code
a {
    color: black;
}
~~~

````css   <-------- if code wrapped with 4 `, the code
a {   will be injected to this page
    color: black;
}
````
```

An example in reStructuredText:

```
Felix Felicis
================

:date: 2012-12-12 12:12
:tags: python, blog, web

Hello World! Welcome to the Felix Felicis World.

::

    /* normal code */
    a {
        color: black;
    }

.. sourcecode:: css

    /* hightlight code */

    a {
        color: black;
    }
```

### 1.3.2 Meta

Metadata that Felix Felicis supports natively:

- date
- public – default is `true`, if set to `false`, this post won’t be included in archive
```

- tags – tags are seprated by comma
- category
- summary
- folder – relative filepath, for example `/home/user/blog/content/a/a.md`, folder will be `a`
- author – see *Multiple Authors* for detail
- template – see *Template* for detail

which means you can access them in template with a shortcut, for example: `{{post.tags}}`.

Metadata that Felix Felicis created itself:

- filepath
- clean_filepath
- filename
- clean_title – https://github.com/lepture/liquidluck/issues/32
- updated

### 1.3.3 Page

Page is the same as post, except that post contains date, page doesn't, post follows permalink, page doesn't.

A example of page in Markdown:

```
# Hello Page

- tags: python, web         <----------- page has no date

----------------

Hello Page

```python
def hello():
    print("Hello Page")
```
```

Page doesn't have a `date`, but it may contain some metadata.

Where will the page be rendered? For example, the path of the page:

```
content/                <-------- source directory
    page1.md
    a_folder/
        page2.md
```

and it will be rendered to:

```
deploy/                 <-------- output directory
    page1.html
    a_folder/
        page2.html
```

It will ignore the `site.prefix`, and therefore, if your settings:

```
site = {
    'name': '...',
    ...
    'prefix': 'blog',
}
```

and you want to you pages to be rendered to `blog` folder, you have to:

```
content/
    blog/               <--------- place your pages under the prefix folder
        page1.md
```

### 1.3.4 File

Any file without a valid markup suffix (e.g. `.md`, `.rst`, `.mkd` ...) is a **File**. It will be copied to the same path:

```
content/
    robots.txt          <--------- this is a file
    media/
        a_pic.jpg       <--------- this is a file
```

And the output will be:

```
deploy/
    robots.txt
    media/
        a_pic.jpg
```

Hence, I suggest that you have a folder named `media`, and you can leave your picture resources there:

```
![alt](/media/a_pic.jpg "title")
```

## 1.4 Configuration

Felix Felicis support **yaml**, **python** and **json** format config file. You can create the config file with:

```
$ liquidluck init
```

### 1.4.1 Overview

The default **python** format config file:

```
# -*- coding: utf-8 -*-
#: settings for liquidluck

#: site information
#: all variables can be accessed in template with ``site`` namespace.
#: for instance: {{site.name}}
site = {
    "name": "Felix Felicis",  # your site name
    "url": "http://lab.lepture.com/liquidluck/",  # your site url
    # "prefix": "blog",
}
```

```
#: this config defined information of your site
#: 1. where the resources  2. how should the site be generated
config = {
    "source": "content",
    "output": "deploy",
    "static": "deploy/static",
    "static_prefix": "/static/",
    "permalink": "{{date.year}}/{{filename}}.html",
    "relative_url": False,
    "perpage": 30,
    "feedcount": 20,
    "timezone": "+08:00",
}


author = {
    "default": "nickname",
    "vars": {}
}

#: active readers
reader = {
    "active": [
        "liquidluck.readers.markdown.MarkdownReader",
        # uncomment to activate rST reader
        # "liquidluck.readers.restructuredtext.RestructuredTextReader",
    ],
    "vars": {}
}

#: active writers
writer = {
    "active": [
        "liquidluck.writers.core.PostWriter",
        "liquidluck.writers.core.PageWriter",
        "liquidluck.writers.core.ArchiveWriter",
        "liquidluck.writers.core.ArchiveFeedWriter",
        "liquidluck.writers.core.FileWriter",
        "liquidluck.writers.core.StaticWriter",
        "liquidluck.writers.core.YearWriter",
        "liquidluck.writers.core.CategoryWriter",
        # "liquidluck.writers.core.CategoryFeedWriter",
        # "liquidluck.writers.core.TagWriter",
        # "liquidluck.writers.core.TagCloudWriter",
    ],
    "vars": {
        # uncomment if you want to reset archive page
        # "archive_output": "archive.html",
    }
}

#: theme variables
theme = {
    "name": "default",

    # theme variables are defined by theme creator
    # you can access theme in template with ``theme`` namespace
    # for instance: {{theme.disqus}}
```

```
    "vars": {
        #"disqus": "your_short_name",
        #"analytics": "UA-21475122-1",
    }
}

#: template variables
template = {
    "vars": {},
    "filters": {},
}
```

### 1.4.2 Permalink

Default permalink style is:

```
{{date.year}}/{{filename}}.html

# output example
tech/intro-of-liquidluck.html
```

There are other permalink styles you may like:

- `{{filename}}.html`
- `{{folder}}/{{filename}}.html`
- `{{category}}/{{filename}}.html`
- `{{date.year}}/{{filename}}.html`
- `{{date.year}}/{{date.month}}/{{filename}}.html`

You can define other keywords in your post, and take them as a part of the permalink:

```
# Hello World

- date: 2012-12-12
- topic: life

----------

content here
```

And then you can set your permalink as: `{{topic}}/{{filename}}.html`. Learn more about *Meta*.

If you don't like `.html` as a part of the permalink, you can set your permalink as:

```
{{category}}/{{filename}}

# or with a slash
{{category}}/{{filename}}/

# slash without server helper
{{category}}/{{filename}}/index.html
```

In this case, you need to make some config of your server, so that everything will be ok. A good example of nginx conf for slash style permalink: nginx.conf.

Issues about permalink:

- https://github.com/lepture/liquidluck/issues/21

### 1.4.3 Multiple Authors

If your site has multiple authors, you can add them to your settings:

```
author = {
    'default': 'lepture',

    'vars': {
        'lepture': {
            'name': 'Hsiaoming Yang',
            'website': 'http://lepture.com',
            'email': 'lepture@me.com',
        },
        'kitty': {
            'name': 'Hello Kitty',
            'website': 'http://hellokitty.com',
        }
    }
}
```

And when you write a post, the default author is 'lepture', but you can change it by:

```
# Hello World

- date: 2012-12-12
- author: kitty

--------

content here
```

Access the author information in template as `{{post.author.name}}` and `{{post.author.website}}`.

For more information on template or theme design, head over to *Theme* section.

The default theme doesn't show any information of the author, it is designed for personal blogging.

### 1.4.4 Readers

There are two readers in Felix Felicis, one is Markdown, and the other is reStructuredText.

#### Customize Reader

Issues that contain information on readers:

- https://github.com/lepture/liquidluck/issues/26

#### Reader Variables

Issues that contain information on readers variables:

- https://github.com/lepture/liquidluck/issues/25

### 1.4.5 Writers

There are many writers in Felix Felicis, and you can add more. If you want to add your own writer to Felix Felics, head over to *Development*.

#### Writers Variables

Every writer can define its own variable, for example the archive write, if you set:

```
writer = {
    'vars': {
        'archive_output': 'archive.html',
    }
}
```

The archive page will be write to **archive.html** instead of **index.html**.

Available writers variables (but you won't need to change them):

- post_template (post.html)
- page_template (page.html)
- archive_template (archive.html)
- **archive_output** (index.html)
- archive_feed_template (feed.xml)
- year_template (archive.html)
- tag_template (archive.html)
- category_template (archive.html)
- category_feed_template (feed.xml)

### 1.4.6 Useful Issues

- https://github.com/lepture/liquidluck/issues/25
- https://github.com/lepture/liquidluck/issues/26
- https://github.com/lepture/liquidluck/issues/30
- https://github.com/lepture/liquidluck/issues/32
- https://github.com/lepture/liquidluck/issues/34

## 1.5 Design Pattern

The design pattern of Felix Felicis contains one significant principle:

> Don't create anything.

That means we didn't (and will not) add any invalid markup like Jekyll and others do. Everything should be valid in its markup language.

And hence there will be no plugin or extension like:

```
{% gist id %}
```

Just like the zen of Python:

> Beautiful is better than ugly.
>
> Nothing is better than everything.

## 1.6 Theme

Template engine of Felix Felicis (liquidluck) is Jinja. It would be great if you have a little knowledge on Jinja Template. The basic syntax is simple, you should know them.

You can learn how to design your own theme by demo:

- https://github.com/lepture/liquidluck-theme-moment
- https://github.com/lepture/liquidluck-theme-octopress

Please create your repo at github with `liquidluck-theme-` prefix. Remember to submit your theme at Theme Gallery.

Get all themes:

```
$ liquidluck search
```

Install a theme:

```
$ liquidluck install moment
```

Install a theme to the global theme gallery:

```
$ liquidluck install moment -g
```

### 1.6.1 Structure

A glance of a theme:

```
your_theme/
    settings.py                <---- theme variables
    filters.py                 <---- filters defined by theme
    static/                    <---- static files
        style.css
        ...
    templates/                 <---- template files
        archive.html
        post.html
        page.html
```

You don't need to copy a `feed.xml` file. Only `archive.html`, `post.html` and `page.html` are required. But you can add more.

### 1.6.2 Template

Sometimes, you don't need to create a total new theme, you just want to make some changes.

For example, you are using the default theme, which means in your settings:

```
theme = {
    'name': 'default'
}
```

You want to make some changes on the post page (like adding readability), in your blog directory, create a post.html template:

```
your_blog/
    settings.py
    content/
    _templates/
        post.html
```

And edit this post.html:

```
{% extends "layout.html" %}

{% block title %}{{post.title}} - {{site.name}}{% endblock %}

{% block main %}
<div class="hentry">
    <h1 class="entry-title">{{post.title}}</h1>
    <time class="updated" datetime="{{post.date|xmldatetime}}">{{post.date.strftime('%Y-%m-%d')}}</ti
    {% if template.readability %}
    <div class="rdbWrapper" data-show-read="1" data-show-send-to-kindle="1"></div>
    <script type="text/javascript" src="http://www.readability.com/embed.js" async></script>
    {% endif %}

    <div class="entry-content">
        {{post.content}}
    </div>
    <div class="entry-tags">
        {% for tag in post.tags %}
        <a href="{{ content_url(site.prefix, 'tag', tag, 'index.html') }}">{{tag}}</a>
        {% endfor %}
    </div>

    {% if theme.disqus %}
    <div id="disqus_thread"></div>
    <script type="text/javascript">
        var disqus_shortname = '{{theme.disqus}}';
        var disqus_title = '{{post.title}}';
        (function() {
            var dsq = document.createElement('script'); dsq.type = 'text/javascript'; dsq.async = tru
            dsq.src = 'http://' + disqus_shortname + '.disqus.com/embed.js';
            (document.getElementsByTagName('head')[0] || document.getElementsByTagName('body')[0]).ap
        })();
    </script>
    {% endif %}
</div>
{% endblock %}
```

And edit your settings, enable readability:

```
template = {
    'vars': {
        'readability': True,
    }
}
```

### 1.6.3 Variables

There are two levels of variables, global and templatable. Global means that this variable can be accessed in every template, and templatable means that this variable can be accessed in specify template.

#### Global Variables

- system, this is all about Felix Felicis:

  ```
  {
      'name': 'Felix Felicis',
      'version': '....',
      'homepage': '....',
      'time': '....',
  }
  ```

  When you create your own theme, you should add copyright of Felix Felicis by:

  ```
  Powered by <a href="{{system.homepage}}">{{system.name}}</a> {{system.version}}
  ```

  `{{system.time}}` means current utc time.

- site, you defined in your settings file:

  ```
  site = {
      'name': "Kitty's BLog",
      'url': 'http://www.example.com',
  }
  ```

- theme, theme variable is defined by theme creator in the theme settings, and users can overwrite theme in blog settings `theme_variables`.

  For example, in the default theme's settings, we have:

  ```
  navigation = [
      {'title': 'Home', 'link': '/'},
      {'title': 'About', 'link': '/about.html'},
  ]
  ```

  Users can rewrite it in blog settings:

  ```
  theme = {
      'vars': {
          'navigation': [
              {'title': 'Home', 'link': '/'},
              {'title': 'Life', 'link': '/life/'},
              {'title': 'Work', 'link': '/work/'},
          ]
      }
  }
  ```

- template, template variable is defined by users in settings with:

  ```
  template = {
      'vars': {
          'readability': True,
      }
  }
  ```

  And it can be access in template by `{{template.readability}}`, this is very useful.

- writer, this variable tells you which writer is rendering this page now:

```
{
    'class': 'ArchiveWriter',
    'name': 'archive',
    'filepath': 'path/to/file.html',
}
```

#### Templatable Variables

Templatable variables are only accessed in specify templates.

- pagination, available in `archive.html`
- post, available in `post.html` and `page.html`

### 1.6.4 Resource Variables

This variable is powerful, for example, `{{resource.posts}}` contains all your public posts. It is related to a writer.

- {{resource.posts}}
- {{resource.pages}}
- {{resource.year}}: if you enabled YearWriter
- {{resource.category}}: if you enabled CategoryWriter
- {{resource.tag}}: if you enabled TagWriter

#### Functions

- content_url
- static_url

### 1.6.5 Filters

Filter is an important concept in Jinja Template.

#### Default Filters

- xmldatetime
- permalink, `{{post|permalink}}` to create the permalink of a post
- tag_url
- year_url
- feed_updated

#### Theme Filters

### 1.6.6 Contributors

If you have designed a theme, you can submit it to the Theme Gallery

## 1.7 Development

Want to contribute to this project? Have no idea about how to read the code? And this section will tell you how to understand it.

### 1.7.1 Command line interface

You should start here, it would be much easier to understand the whole things. Take a look at `cli.py`.

### 1.7.2 Readers

### 1.7.3 Writers

### 1.7.4 Namespace

### 1.7.5 Utilities

## 1.8 Goodies

Gifts that makes Felix Felicis easy to use.

### 1.8.1 Preview Server

Preview your blog with:

```
$ liquidluck server
$ liquidluck server -p 8888
$ liquidluck server -p 8888 -s settings.py
```

Preview server now support livereload, when you are editing a post, it will auto compile and auto refresh the browser for you. **Added in version 1.12**

To enable livereload, your should install tornado:

```
$ pip install tornado
```

### 1.8.2 Oh My Zsh Plugin

If you are using oh-my-zsh, there is a plugin for you.

https://github.com/lepture/liquidluck/tree/master/oh-my-zsh-plugins

Copy `liquidluck` to `~/.oh-my-zsh/plugins`:

```
$ cp -r oh-my-zsh-plugins/liquidluck ~/.oh-my-zsh/plugins/
```

Then edit your zshrc file:

```
plugins=(git ... liquidluck)
```

Now you can tab complete every Felix Felicis command:

```
$ liquidluck b(tab)
$ liquidluck build (tab)
```

### 1.8.3 Webhook

Felix Felicis supports webhook since v1.6. When you push to GitHub or BitBucket, your blog can generate itself.

First, you need to install Felix Felicis on your server:

```
$ pip install liquidluck
```

Second, your blog repo on your server:

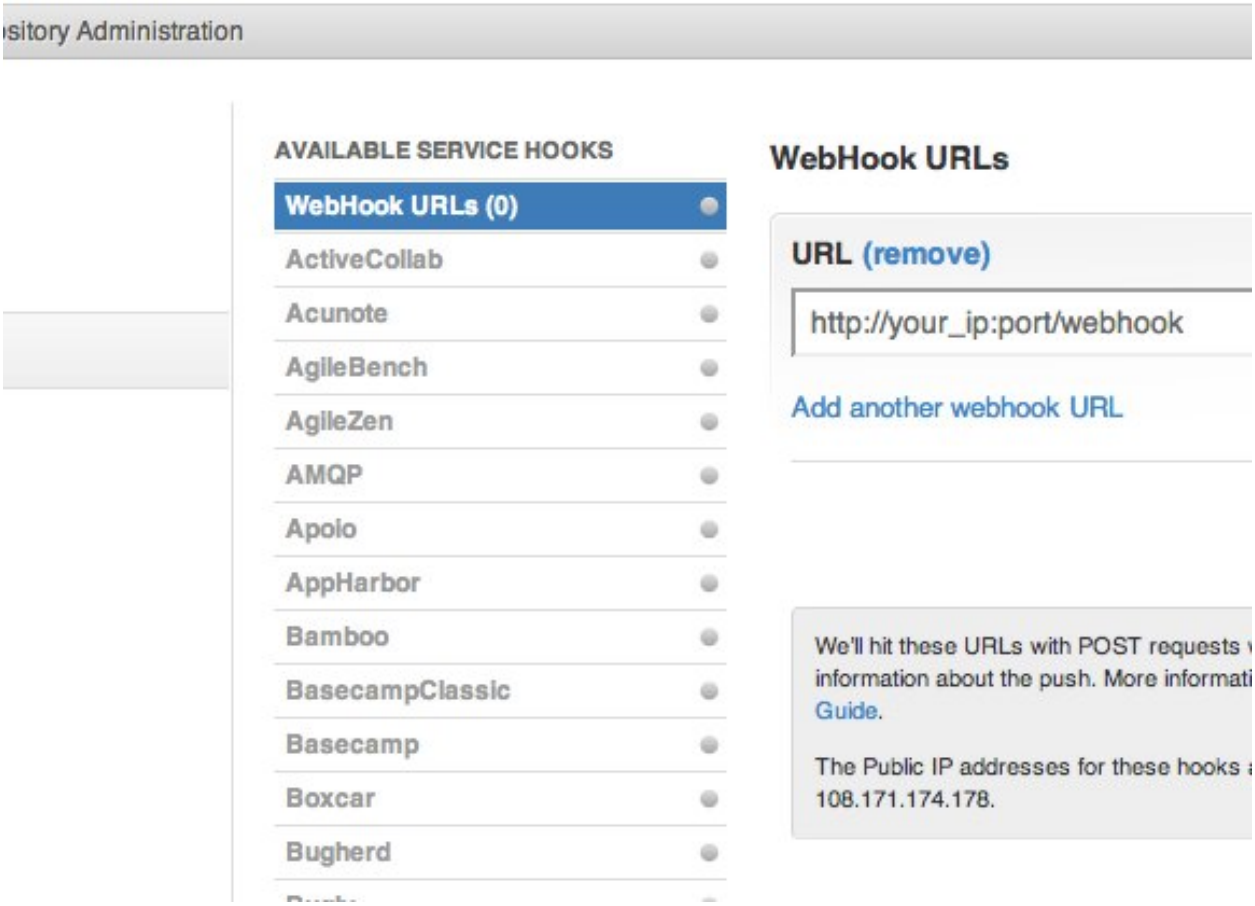
```
$ git clone git://path/to/your/repo blog
```

Then, start webhook daemon in your blog:

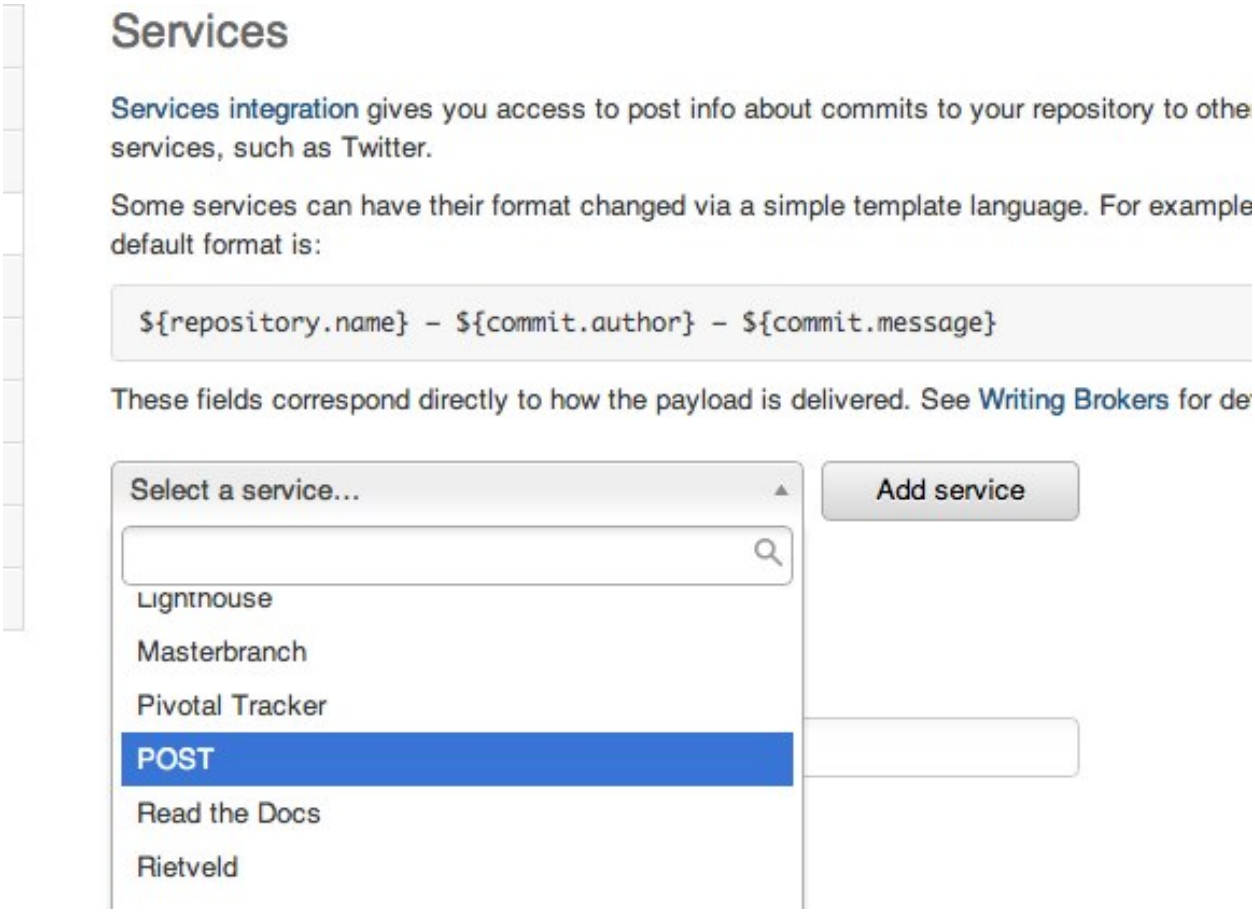
```
$ cd blog
$ liquidluck webhook start -p 9876
```

Configure webhook on GitHub or BitBucket. We take GitHub as an example.

Head over to your blog source repo admin, select **Service Hooks**




If you prefer BitBucket, you should select the POST service:




If your server ip is 88.88.88.88, you can add a URL:

`http://88.88.88.88:9876/webhook`

And when you push to GitHub, your server will update the repo and generate the whole site.

## 1.9 Changelog

All history since the new Felix Felicis are listed here:

### 1.9.1 Version 3.7

Released on Nov 9th, 2012

- code improvement (not bugfix)
- add option quiet for `liquidluck build`
- add option output for `liquidluck build`
- livereload server more reliable

### 1.9.2 Version 3.6

Released on Nov 2nd, 2012

- `clean_folder` is replaced by `folder`
- `clean_filepath` is replaced by `relative_filepath`
- delete `Post.embed_author`
- livereload server work again
- add `resource.files` params

### 1.9.3 Version 3.5

Released on Oct 31th, 2012

- fix `liquidluck search` unicode bug #57
- add theme.debug vars for preview server

### 1.9.4 Version 3.4

Released on Sep 20th, 2012

- **ATTENTION**: settings.py in themes should be named as theme.py now
- install and update theme
- logging improvement

### 1.9.5 Version 3.3

Released on Sep 17th, 2012

- support for [[]]
- updates on default theme

### 1.9.6 Version 3.2

Released on Sep 13rd, 2012

- bugfix for relative url
- support for `

### 1.9.7 Version 3.1

Released on Sep 12nd, 2012

- bugfix
- change server host to 127.0.0.1

### 1.9.8 Version 3.0

Released on Sep 12nd, 2012

- new config format: yaml, python, json
- redesigned

### 1.9.9 Version 2.0

Released on Sep 7th, 2012

- support for relative url
- support for inject html, css, javascript
- bugfix for server
- code structure changed
- github search api changed

### 1.9.10 Version 1.14

Released on Oct 23th, 2012

- add render params: writer
- API changed. `liquidluck.readers.base.Post`, delete `filedir`, add `clean_filepath`
- force search theme from internet

### 1.9.11 Version 1.13

Released on Jul 16th, 2012

- fix markdown meta parser
- webhook deamon enhancement

### 1.9.12 Version 1.12

Released on Jul 9th, 2012

- LiveReload Server
- GitHub Search API fix
- docutils is optional

### 1.9.13 Version 1.11

Released on Jun 20th, 2012

- fix permalink filter, support {{filename}}/index.html now. #41
- update default theme
- improve command line interface. #43

### 1.9.14 Version 1.10

Released on Jul 17th, 2012

- improve on feed render #40
- config feed output tags
- server bugfix
- built in filters of tag_url and year_url

### 1.9.15 Version 1.9

Released on Jul 4th, 2012

- improve server, can be used as a standalone app with autoindex support
- default permalink changed to {{date.year}}/{{filename}}
- timezone fix
- update theme

### 1.9.16 Version 1.8

Released on Jul 1st, 2012

- search theme from github
- timezone support

### 1.9.17 Version 1.7

Released on Jun 29th, 2012

- webhook supports submodule
- webhook supports hg
- preview server #35

### 1.9.18 Version 1.6

Released on Jun 29th, 2012

- webhook support #33
- add clean_title #32
- table support in markdown

### 1.9.19 Version 1.5

Released on Jun 28th, 2012.

- bugfix for `static_url` encoding error
- command line interface changed #31
- update the default theme

### 1.9.20 Version 1.4

Released on Jun 25th, 2012.

- add TagCloudWriter
- bugfix #24 #29

### 1.9.21 Version 1.3

Released on Jun 21th 2012.

- customize markdown link transform
- customize post class
- add filedir property for post

### 1.9.22 Version 1.2

Released on Jun 19th 2012.

- site['prefix'] configuration

### 1.9.23 Version 1.1

Released on Jun 19th 2012.

- search and install theme available
- bugfix issue#20

### 1.9.24 Version 1.0

Released on Jun 16th 2012. The new Felix Felicis.

CHAPTER 2

# Reporting bugs

We keep an issue tracker at GitHub, where you can report a bug by opening a new issue. If you want any help, please contact lepture@me.com; English and Chinese are accepted.