**GUIA PARA CRIAÇÃO DE UM FRAMEWORK WEB EM PYTHON**

<https://www.oreilly.com/library/view/python-web-frameworks/9781492037873/ch04.html>

**PARTES DE UM FRAMEWORK WSGI BÁSICO**

Se você decidir construir um framework, usar Werkzeug ou WebOb é uma boa recomendação. Você deseja se concentrar nas partes configuráveis ​​pelo usuário, não nas partes internas.

**ROTEAMENTO**

Você precisará de alguma forma de corresponder um URL a uma view. A maneira mais comum de fazer isso é usando regular expression routes, como Django ou Flask. Se você quiser adotar essa abordagem, considere usar o Werkzeug, que inclui um sistema de roteamento pronto para uso. Existem também algumas bibliotecas de roteamento disponíveis no PyPI que podem ser usadas com WebOb ou sozinhas.

Existem outras maneiras de mapear URLs para views. O Pyramid oferece travessia, que trata o conteúdo do seu site como uma árvore e mapeia views para recursos ou tipos de recursos. CherryPy usa métodos expostos de classes Python regulares. Talvez você possa pensar em outras abordagens. Basta usar algo que pareça natural para você.

**TEMPLATES**

Existem alguns sistemas de templates muito populares para Python e é altamente recomendável usar um deles. Jinja2 e Mako são duas opções muito boas.

Alguns frameworks são altamente adaptados para funcionar com um sistema de template escolhido, enquanto outras deixam essa decisão completamente nas mãos do desenvolvedor. Seja qual for a sua escolha, você não pode errar, pois a maioria desses sistemas de modelos são comprovados e muito estáveis.

**OUTRAS CARACTERÍSTICAS**

A maioria dos frameworks oferece outros recursos como parte do pacote. A autenticação é uma necessidade comum e, portanto, geralmente é fornecida de alguma forma. Existem muitos sistemas de autenticação e bibliotecas por aí. O melhor conselho aqui é tentar ser flexível. Você nunca sabe como um usuário precisará autenticar seus usuários.

Muitos frameworks pressupõem (talvez com menos certeza atualmente) que um back-end de banco de dados relacional fará parte de uma aplicação web e oferecerá pelo menos algum grau de suporte a banco de dados. Se essa for sua decisão, basta optar pela integração SQLAlchemy. É a solução mais completa (e popular) para Python.

Considere oferecer algum suporte para manipulação de formulários e validação de entrada, mas esteja ciente de que existem muitas bibliotecas que fazem isso muito bem. Pesquise um pouco suas alternativas antes de decidir fazer você mesmo.

Se você pretende que seu framework seja usado por outras pessoas, ou pelo menos planeja usá-lo mais de uma vez, você precisa oferecer um sistema de configuração que permita que seus usuários configurem e implantem facilmente seus aplicativos.

**DOCUMENTAÇÃO**

Mesmo que seu framework seja apenas para uso interno da sua empresa, faça o possível para documentá-la. Na verdade, se você não consegue documentar, é melhor usar um dos frameworks existentes. Os futuros mantenedores de seus aplicativos agradecerão.

Uma boa documentação consiste em muito mais do que nomes de classes e métodos. Alguma documentação narrativa é muito útil e muitas pessoas acham que os exemplos de código falam por si. Iniciantes gostam de tutoriais e explicações menos técnicas. Use uma ferramenta de documentação, como o Sphinx, para facilitar a geração de documentação.

**BLOCOS DE CONSTRUÇÃO DO FRAMEWORK**

Você não precisa começar seu framework do zero. Existem alguns kits de ferramentas WSGI que cuidam do básico, permitindo que você se concentre nos recursos exclusivos fornecidos pelo seu framework.

Os dois kits de ferramentas WSGI mais populares são WebOb e Werkzeug. Muitos frameworks web usam uma dessas bibliotecas. Eles são testados ao longo do tempo e possuem todos os recursos, portanto fornecem uma excelente base para o desenvolvimento de aplicativos da web sem se preocupar com os detalhes de HTTP e WSGI.

**WebOb (http://webob.org)**

WebOb fornece objetos que mapeiam grande parte do comportamento especificado do HTTP, incluindo análise de cabeçalho, negociação de conteúdo e manipulação correta de solicitações condicionais e de intervalo. Pyramid é uma estrutura bem conhecida que usa WebOb.

**Werkzeug (http://werkzeug.pocoo.org)**

Werkzeug é um dos módulos utilitários WSGI mais avançados. Inclui um depurador poderoso, objetos de solicitação e resposta completos, utilitários HTTP para lidar com tags de entidade, cabeçalhos de controle de cache, datas HTTP, manipulação de cookies, uploads de arquivos e um poderoso sistema de roteamento de URL. Flask, um dos frameworks web Python mais populares, usa Werkzeug.

**WSGI**

WSGI, que significa "Web Server Gateway Interface", é uma especificação padrão em Python que define uma interface entre servidores web e aplicações web escritas em Python. Essa interface tem o objetivo de facilitar a interoperabilidade entre servidores web e frameworks web em Python, permitindo que diferentes servidores web se comuniquem de maneira uniforme com as aplicações web.

A principal ideia por trás do WSGI é separar a lógica da aplicação web do servidor web. Isso significa que você pode escrever sua aplicação web de maneira independente do servidor web específico no qual ela será executada. Desse modo, as aplicações web escritas de acordo com a especificação WSGI podem ser implantadas em diferentes servidores web compatíveis com WSGI sem a necessidade de modificação do código da aplicação.

**ALGUNS RECURSOS ÚTEIS**

Se você decidir escrever seu próprio framework, ou apenas quiser saber um pouco mais sobre como fazê-lo, existem alguns excelentes recursos da web que podem lhe mostrar como. Aqui estão três dos melhores.

Você pode começar com A Do-It-Yourself Framework, no qual Ian Bicking explica o que é WSGI construindo uma estrutura simples. Este é um tutorial muito bom e uma ótima maneira de descobrir o que está dentro de uma estrutura real.

Ian Bicking tem um segundo tutorial, desta vez usando a biblioteca WebOb. Another Do-It-Yourself Framework tem mais detalhes sobre as partes de um framework e até explica linha por linha o código.

Algumas pessoas aprendem melhor usando screencasts em vez de documentação escrita. Se for esse o seu caso, você achará útil a série de screencasts de Chris McDonough sobre o uso de repoze.bfg para construir uma microestrutura. Lembre-se, repoze.bfg é o framework que se tornou o Pyramid. Mesmo que você não use nenhuma dessas estruturas, você ainda achará esta série muito instrutiva. Os vídeos estão em <http://bfg.repoze.org/videos>.