Adding HTTP Method Overrides

Some HTTP proxies do not support arbitrary HTTP methods or newer HTTP methods (such as PATCH). In that case it's possible to "proxy" HTTP methods through another HTTP method in total violation of the protocol.

The way this works is by letting the client do an HTTP POST request and set the `X-HTTP-Method-Override` header. Then the method is replaced with the header value before being passed to Flask.

This can be accomplished with an HTTP middleware:

```
class HTTPMethodOverrideMiddleware(object):
    allowed_methods = frozenset([
        'GET',
        'HEAD',
        'POST',
        'DELETE',
        'PUT',
        'PATCH',
        'OPTIONS'
    ])
    bodyless_methods = frozenset(['GET', 'HEAD', 'OPTIONS', 'DELETE'])

    def __init__(self, app):
        self.app = app

    def __call__(self, environ, start_response):
        method = environ.get('HTTP_X_HTTP_METHOD_OVERRIDE', '').upper()
        if method in self.allowed_methods:
            environ['REQUEST_METHOD'] = method
        if method in self.bodyless_methods:
            environ['CONTENT_LENGTH'] = '0'
        return self.app(environ, start_response)
```

To use this with Flask, wrap the app object with the middleware:

```
from flask import Flask

app = Flask(__name__)
app.wsgi_app = HTTPMethodOverrideMiddleware(app.wsgi_app)
```