GOOGLE APP ENGINE

CHAPTER 4

PHP: Handling Requests

Handling Requests

Requests and domains

App Engine determines that an incoming request is intended for your application using the domain name of the request. A request whose domain name is `http://your_app_id.appspot.com` is routed to the application whose ID is `your_app_id`. Every application gets an `appspot.com` domain name for free.

`appspot.com` domains also support subdomains of the form `subdomain-dot-your_app_id.appspot.com`, where `subdomain` can be any string allowed in one part of a domain name (not `.`). Requests sent to any subdomain in this way are routed to your application.

You can set up a custom top-level domain using Google Apps. With Google Apps, you assign subdomains of your business's domain to various applications, such as Google Mail or Sites. You can also associate an App Engine application with a subdomain. For convenience, you can set up a Google Apps domain when you register your application ID, or later from the Administrator Console. See Deploying your Application on your Google Apps URLfor more information.

Requests for these URLs all go to the version of your application that you have selected as the default version in the Administration Console. Each version of your application also has its own URL, so you can deploy and test a new version before making it the default version. The version-specific URL uses the version identifier from your app's configuration file in addition to the `appspot.com` domain name, in this pattern: `http://version_id-dot-latest-dot-your_app_id.appspot.com` You can also use subdomains with the version-specific URL:`http://subdomain-dot-version_id-dot-latest-dot-your_app_id.appspot.com`

The domain name used for the request is included in the request data passed to the application. If you want your app to respond differently depending on the domain name used to access it (such as to restrict access to certain domains, or redirect to an official domain), you can check the request data (such as the `Host` request header) for the domain from within the application code and respond accordingly.

Please note that in April of 2013, Google stopped issuing SSL certificates for double-wildcard domains hosted at `appspot.com` (i.e. `*.*.appspot.com`). If you rely on such URLs for HTTPS access to your application, please change any application logic to use "-dot-" instead of ".". For example, to access version "1" of application "myapp" use "https://1-dot-myapp.appspot.com" instead of "https://1.myapp.appspot.com." If you continue to use "https://1.myapp.appspot.com" the certificate will not match, which will result in an error for any User-Agent that expects the URL and certificate to match exactly.

Requests

When App Engine receives a web request for your application, it calls the handler script that corresponds to the URL, as described in the application's app.yaml configuration file .

App Engine runs multiple instances of your application, each instance has its own web server for handling requests. Any request can be routed to any instance, so consecutive requests from the same user are not necessarily sent to the same instance. The number of instances can be adjusted automatically as traffic changes.

The server determines which PHP handler script to run by comparing the URL of the request to the URL patterns in the app's configuration file. It then runs the script populated with the request data. The server puts the request data in environment variables and the standard input stream. The script performs actions appropriate to the request, then prepares a response and puts it on the standard output stream.

Request headers

An incoming HTTP request includes the HTTP headers sent by the client. For security purposes, some headers are sanitized or amended by intermediate proxies before they reach the application.

The following headers are removed from the request:

- `Accept-Encoding`
- `Connection`
- `Keep-Alive`
- `Proxy-Authorization`
- `TE`
- `Trailer`
- `Transfer-Encoding`

In addition, the header `Strict-Transport-Security` is removed from requests served to any domains other than `appspot.com` or `*.appspot.com`.

These headers relate to the transfer of the HTTP data between the client and server, and are transparent to the application. For example, the server may automatically send a gzipped response, depending on the value of the`Accept-Encoding` request header. The application itself does not need to know which content encodings the client can accept.

Note: Entity headers (headers relating to the request body) are not sanitized or checked, so applications should not rely on them. In particular, the `Content-MD5` request header is sent unmodified to the application, so may not match the MD5 hash of the content. Also, the `Content-Encoding` request header is not checked by the server, so if the client sends a gzipped request body, it will be sent in compressed form to the application.

As a service to the app, App Engine adds some headers:

`X-AppEngine-Country`

Country from which the request originated, as an ISO 3166-1 alpha-2 country code. App Engine determines this code from the client's IP address.

`X-AppEngine-Region`

Name of region from which the request originated. This value only makes sense in the context of the country in `X-AppEngine-Country`. For example, if the country is "US" and the region is "ca", that "ca" means "California", not Canada.

`X-AppEngine-City`

Name of the city from which the request originated. For example, a request from the city of Mountain View might have the header value `mountain view`.

`X-AppEngine-CityLatLong`

Latitude and longitude of the city from which the request originated. This string might look like "37.386051,-122.083851" for a request from Mountain View.

Responses

App Engine calls the script with the `$_REQUEST` array populated, buffers any output from the script, and when the script completes execution, sends the buffered output to the end user.

App Engine does not support sending data to the client, performing more calculations in the application, then sending more data. In other words, App Engine does not support "streaming" data in response to a single request.

Dynamic responses are limited to 32MB. If a script handler generates a response larger than this limit, the server sends back an empty response with a 500 Internal Server Error status code. This limitation does not apply to responses that serve data from Google Cloud Storage .

If the client sends HTTP headers with the request indicating that the client can accept compressed (gzipped) content, App Engine compresses the response data automatically and attaches the appropriate response headers. It uses both the `Accept-Encoding` and `User-Agent` request headers to determine if the client can reliably receive compressed responses. Custom clients can indicate that they are able to receive compressed responses by specifying both `Accept-Encoding` and `User-Agent` headers with a value of `gzip`. The`Content-Type` of the response is also used to determine whether compression is appropriate; in general, text-based content types are compressed, whereas binary content types are not.

Note: In the absence of a `Content-Type` header provided by your application, a default `Content-Type: text/html` header will be set, but the automatic gzip compression will not be applied.

The following headers are ignored and removed from the response:

- `Connection`
- `Content-Encoding`
- `Content-Length`
- `Date`
- `Keep-Alive`
- `Proxy-Authenticate`
- `Server`
- `Trailer`
- `Transfer-Encoding`
- `Upgrade`

In addition, the header `Strict-Transport-Security` is removed from responses served from any domains other than `*.appspot.com`.

Headers with non-ASCII characters in either the name or value are also removed. In addition, the following headers are added or replaced in the response:

`Cache-Control`, `Expires` and `Vary`

These headers specify caching policy to intermediate web proxies (such as Internet Service Providers) and browsers. If your script sets these headers, they will usually be unmodified, unless the response has a Set-Cookie header, or is generated for a user who is signed in using an administrator account. Static handlers will set these headers as directed by the configuration file. If you do not specify a `Cache-Control`, the server may set it to `private`, and add a `Vary: Accept-Encoding` header.

If you have a Set-Cookie response header, the `Cache-Control` header will be set to `private` (if it is not already more restrictive) and the `Expires` header will be set to the current date (if it is not already in the past). Generally, this will allow browsers to cache the response, but not intermediate proxy servers. This is for security reasons, since if the response was cached publicly, another user could subsequently request the same resource, and retrieve the first user's cookie.

Note: The development server disables caching by default (for both script and static content), so that if you do not specify caching headers, you will not have cached content while developing your app.

`Content-Encoding`

Depending upon the request headers and response `Content-Type`, the server may automatically compress the response body, as described above. In this case, it adds a `Content-Encoding: gzip`header to indicate that the body is compressed.

`Content-Length` or `Transfer-Encoding`

The server always ignores the `Content-Length` header returned by the application. It will either set`Content-Length` to the length of the body (after compression, if compression is applied), or delete`Content-Length`, and use chunked transfer encoding (adding a `Transfer-Encoding: chunked`header).

`Content-Type`

If not specified by the application, the server will set a default `Content-Type: text/html` header.

`Date`

Set to the current date and time.

`Server`

Set to `Google Frontend`. The development server sets this to `Development/x`, where *x* is the version number.

If you access your site while signed in using an administrator account, App Engine includes per-request statistics in the response headers:

`X-AppEngine-Estimated-CPM-US-Dollars`

An estimate of what 1,000 requests similar to this request would cost in US dollars.

`X-AppEngine-Resource-Usage`

The resources used by the request, including server-side time as a number of milliseconds.

Responses with resource usage statistics will be made uncacheable.

The request timer

A PHP script has a limited amount of time to generate and return a response to a request, typically around 60 seconds. Once the deadline has been reached, the `TIMEOUT` bit on the connection status bitfield is set. Your script will then have a short second deadline to clean up any long running tasks and return a response to the user.

```
function check_conn_timeout() {

  $status = connection_status();

  if (($status & CONNECTION_TIMEOUT) == CONNECTION_TIMEOUT) {

    echo 'Got timeout';

  }

}

while(1) {

  check_conn_timeout();

  sleep(1);

}
```

If your script hasn't returned a response by the second deadline, the handler is terminated and a default error response is returned.

While a request can take as long as 60 seconds to respond, App Engine is optimized for applications with short-lived requests, typically those that take a few hundred milliseconds. An efficient app responds quickly for the majority of requests. An app that doesn't will not scale well with App Engine's infrastructure.

SPDY

App Engine applications will automatically use the SPDY protocol when accessed over SSL by a browser that supports SPDY. This is a replacement for HTTP designed by Google and intended to reduce the latency of web page downloads. The use of SPDY should be entirely transparent to both applications and users (applications can be written as if normal HTTP was being used). For more information, see the SPDY project page.

Logging

The App Engine web server captures everything the handler script writes to the standard output stream for the response to the web request. It also captures everything the handler script writes to the standard error stream, and stores it as log data. Log data for your application can be viewed and analyzed using the Administration Console, or downloaded using appcfg.py request_logs.

The App Engine PHP runtime environment includes support for logging arbitrary messages from your application using PHP's built-in `syslog()` function, which invokes the Logs PHP Api.

```
if (authorized_user()) {

  // Some code

} else {

  syslog(LOG_WARNING, 'Unauthorized access');

}
```

App caching

The PHP runtime environment includes the Alternative PHP Cache (APC) which can cache PHP intermediate code and significantly improve your application's response time. You can disable APC opcode caching by setting`apc.enabled = "0"` in the application's php.ini file.

Quotas and limits

Google App Engine automatically allocates resources to your application as traffic increases. However, this is bound by the following restrictions:

- App Engine reserves automatic scaling capacity for applications with low latency, where the application responds to requests in less than one second. Applications with very high latency (over one second per request for many requests) and high throughput require Silver, Gold, or Platinum support. Customers with this level of support can request higher throughput limits by contacting their support representative.

- Applications that are heavily CPU-bound may also incur some additional latency in order to efficiently share resources with other applications on the same servers. Requests for static files are exempt from these latency limits.

Each incoming request to the application counts toward the Requests limit. Data sent in response to a request counts toward the Outgoing Bandwidth (billable) limit.

Both HTTP and HTTPS (secure) requests count toward the Requests, Incoming Bandwidth (billable), andOutgoing Bandwidth (billable) limits. The Quota Details page of the Admin Console also reports Secure Requests,Secure Incoming Bandwidth, and Secure Outgoing Bandwidth as separate values for informational purposes. Only HTTPS requests count toward these values. See the Quotas page, and the "Quota Details" section of the Admin Console for more information.

In addition to system-wide safety limits, the following limits apply specifically to the use of request handlers:

Limit	Amount
request size	32 megabytes
response size	32 megabytes
request duration	60 seconds
maximum total number of files (app files and static files)	10,000 total 1,000 per directory
maximum size of an application file	32 megabytes
maximum size of a static file	32 megabytes
maximum total size of all application and static files	first 1 gigabyte is free $ 0.026 per gigabyte per month after first 1 gigabyte