Health check standard

Document history

Date	Version	Notes
31-Jan-2013	0.1	Initial version
1-Mar-2013	0.2	Shared with TomA
8-May-2013	0.9	Added text around checking ALL dependencies, so that it replicates Connection and Dependency checks.
28-Aug-2013	0.95	Revised by Andrew Betts
5-Jun-2014	1.0	New Version approved by Architecture Review Board
7-Oct-2015	1.1	Added systemCode field; Added id field to check; Revised panic guide description. Approved by Architecture Strategy Group

Contents

FT web applications (whether or not they have a UI) *must* include a health check capability, which forms an important part of devops integration. Developers are expected to take responsibility for ensuring that their applications are supportable by operations staff when

the developer isn't around, and that it's straightforward to set up and maintain monitoring of the application's current state.

If an application develops a fault, colleagues (developers and ops engineers alike), will look to health checks for direction on what to do, and where to look to resolve the fault. Good, professional, and detailed checks help others to find and resolve faults easily. Health check endpoints can also be used to verify that applications are working correctly in dev or test environments.

Health checks are not intended to be used in routing decisions. Instead, see Good to go. As an example, an application can be considered healthy but unable to accept requests if it has been deliberately deactivated.

Other documents explain the information checks should contain, and mechanisms for alerting via email. The overall monitoring and alerting approach is described in the overview document.

Endpoints

Applications must respond to HTTP requests to the `/__health` path on the same port as the application of every public hostname or IP address on which the service is offered. For example, if the service is available at `foo.webservices.ft.com`, but also works on IP `1.2.3.4`, both `http://foo.webservices.ft.com/__health` and `http://1.2.3.4/__health` must be supported.

The endpoint must be accessible from all FT public IP ranges and internal networks. It is acceptable, though optional, to deny access from other origins (and if access is denied, a 403 Forbidden response *must* be returned).

If the application is served over HTTPS, a health check *must* also be served over HTTPS, and likewise if the application is served over HTTP, a health check *must* also be served over HTTP.

Response

The response status *must* describe the state of the health response according to normal HTTP rules. If the healthcheck response is able to be properly constructed and returned, the response *must* be 200 OK (irrespective of the current status of any checks).

The response *should* provide the following headers, in addition to those required by the HTTP standard:

Header	Value	Why?
Cache-control	no-store	Clients are directed to not cache health check responses.
Content-Length	*Size of response body*	Clients must be able to determine that the response is complete.
Content-Type	application/json;	The output must be JSON and UTF-8.
Date	*RFC 1123 date*	Required by HTTP 1.1. Used as lastUpdated date if

		Last-modified header is not present.
Last-Modified	*RFC 1123 date*	Optional, if present will provide a default for any lastUpdated values that are missing from checks.

Responses *should* avoid using chunked transfer-encoding to provide better compatibility with legacy clients or those with an incomplete implementation of the HTTP protocol.

If the client has been awaiting a response for more than 10 seconds, the application *should* abort the generation of the health check response, and instead return a 503 Service Unavailable response with either of the following in the response body:

- a notice of the timeout; or
- a valid health check response with those checks that timed out marked as failed with an appropriately informative check output field.

The application must not perform checks synchronously that may take more than 10 seconds to run under normal conditions. Checks that require a long runtime must be run asynchronously on a schedule, and the health check endpoint must report the result from the last time the check was run.

Syntax

The response body *must* be valid JSON, encoded as UTF-8, and compliant with the following data model.

Field / structure	Type (required ?)	Description
`{`		
`schemaVersion`	number*	Set to 1*
`systemCode`	string*	A standard FT system code, as defined in the System Code Standard
`name`	string*	Name of application, including any appropriate qualifiers to make it uniquely identifiable. Must match a **service name** defined in CMDBv2 system registry definition.
`description`	string	Description of application for support personnel in no more than 200 chars (good things to include are the end user functionality enabled by the service)
`checks [`	array*	A list of checks performed by this application
`{`	object*	An object representing a single check
`id`	string*	An identifier for this check, unique for the given System Code. Must only consist of lowercase alphanumeric characters and hyphens.
`name`	string*	As defined in Check standard; and unique within this healthcheck.
`ok`	boolean*	Whether the check is currently passing
`severity`	integer*	As defined in Check standard
`businessImpact`	string*	As defined in Check standard

`technicalSummary`	string*	As defined in Check standard
`panicGuide`	string*	As defined in Check standard. A useful panic guide MUST be provided, even when the check is healthy. Panic guides MAY additionally include contextual information depending on the state of the system.
`checkOutput`	string	As defined in Check standard
`lastUpdated`	string	As defined in Check standard. If not present, clients must revert to the value of the Last-Modified or Date HTTP headers (see below).
`},`		
`{ ... }`	object	Repeat for additional checks
`]`		
`}`		

Things to check

Each check *must* use the same code paths and credentials as the code itself. A list of what to monitor is covered by the FT Monitoring Standards document.

Applications *should* include checks for anything else that is relevant to the application's health. For example, running synthetic transactions against themselves.

Interpretation of health check data

Where a check does not have a **`lastUpdated`** date, clients *must* substitute the value of the **`Last-Modified`** HTTP response header (if there is no **`Last-Modified`** header on the response, the client *must* instead use the value from the **`Date`** HTTP response header, which is required to be present by the HTTP spec).

Where the **`lastUpdated`** date of a check is more than 1 hour ago, the client *may* override the **`ok`** parameter of each check, it as **`false`** despite the value being **`true`**.

Monitoring systems *should* poll health pages once per minute, and *must* poll at least once per ten minute period.reading

A failure of a healthcheck URL to deliver a response within 10 seconds *may* be considered a failure of all checks previously published at that endpoint.