

# Pass Kit Web Service Reference

# Contents

## **PassKit Web Service Reference** 3

Registering a Device to Receive Push Notifications for a Pass 3

Getting the Serial Numbers for Passes Associated with a Device 4

Getting the Latest Version of a Pass 5

Unregistering a Device 6

Logging Errors 7

## **Document Revision History** 8

# PassKit Web Service Reference

Companion Guide *Pass Kit Programming Guide*

A REST-style web service protocol is used to communicate with your server about changes to passes, and to fetch the latest version of a pass when it has changed. The endpoints always begin with the web service URL, as specified in the pass, followed by the protocol version number. For example, a request for the latest version of the pass of type `com.apple.pass.example` and serial number `ABC123` might look like the following:

Protocol version number

Serial number

**https://example.com/foo/v1/passes/com.apple.pass.example/ABC123**

Web service URL

Pass type ID

## Registering a Device to Receive Push Notifications for a Pass

POST request to

`https://webServiceURL /version /devices/deviceLibraryIdentifier /registrations/passTypeIdIdentifier /serialNumber`

## Parameters

webServiceURL

The URL to your web service, as specified in the pass.

*version*

The protocol version. Currently, v1.

*deviceLibraryIdentifier*

A unique identifier that is used to identify and authenticate this device in future requests.

*passTypeldentifizier*

The pass's type, as specified in the pass.

*serialNumber*

The pass's serial number, as specified in the pass.

## Header

The Authorization header is supplied; its value is the word “ApplePass,” followed by a space, followed by the pass’s authorization token as specified in the pass.

### Payload

The POST payload is a JSON dictionary, containing a single key and value:

#### *pushToken*

The push token that the server can use to send push notifications to this device.

### Response

- If the serial number is already registered for this device, return HTTP status 200.
- If registration succeeds, return HTTP status 201.
- If the request is not authorized, return HTTP status 401.
- Otherwise, return the appropriate standard HTTP status.

### Discussion

Any time the pass is updated, your server sends a push notification with an empty JSON dictionary as the payload to the device using the given push notification token. This continues until the device is explicitly unregistered (as described in [“Unregistering a Device”](#) (page 6)).

## Getting the Serial Numbers for Passes Associated with a Device

GET request to

`https://webServiceURL /version /devices/deviceLibraryIdentifier /registrations/passTypeIdentifier ?passesUpdatedSince=tag`

### Parameters

#### *webServiceURL*

The URL to your web service, as specified in the pass.

#### *version*

The protocol version. Currently, v1.

#### *deviceLibraryIdentifier*

A unique identifier that is used to identify and authenticate the device.

#### *passTypeIdentifier*

The pass’s type, as specified in the pass.

### *tag*

A tag from a previous request. (*optional*)

If the `passesUpdatedSince` parameter is present, return only the passes that have been updated since the time indicated by `tag`. Otherwise, return all passes.

### Response

- If there are matching passes, return HTTP status 200 with a JSON dictionary with the following keys and values:

`lastUpdated` (string)

The current modification tag.

`serialNumbers` (array of strings)

The serial numbers of the matching passes.

- If there are no matching passes, return HTTP status 204.
- Otherwise, return the appropriate standard HTTP status.

### Discussion

The modification tag is used to give a name to a point in time. It is typically convenient to use a timestamp, but the server is free to use another approach. The tag is treated as an opaque value by the system.

## Getting the Latest Version of a Pass

GET request to `https://webServiceURL/version/passes/passTypeIdentifier/serialNumber`

### Parameters

*webServiceURL*

The URL to your web service, as specified in the pass.

*version*

The protocol version. Currently, v1.

*passTypeIdentifier*

The pass's type, as specified in the pass.

*serialNumber*

The unique pass identifier, as specified in the pass.

### Header

The Authorization header is supplied; its value is the word “ApplePass”, followed by a space, followed by the pass’s authorization token as specified in the pass.

### Response

- If request is authorized, return HTTP status 200 with a payload of the pass data.
- If the request is not authorized, return HTTP status 401.
- Otherwise, return the appropriate standard HTTP status.

### Discussion

Support standard HTTP caching on this endpoint: check for the `If-Modified-Since` header or entity tags, and return HTTP status code 304 if the pass has not changed.

## Unregistering a Device

DELETE request to

`https://webServiceURL/version/devices/deviceLibraryIdentifier/registrations/passTypeIdentifier`

### Parameters

*webServiceURL*

The URL to your web service, as specified in the pass.

*version*

The protocol version. Currently, v1.

*deviceLibraryIdentifier*

A unique identifier that is used to identify and authenticate the device.

*passTypeIdentifier*

The pass’s type, as specified in the pass.

### Header

The Authorization header is supplied; its value is the word “ApplePass”, followed by a space, followed by the pass’s authorization token as specified in the pass.

### Response

- If disassociation succeeds, return HTTP status 200.
- If the request is not authorized, return HTTP status 401.

- Otherwise, return the appropriate standard HTTP status.

### Discussion

The server should disassociate the specified device from the pass, and no longer send push notifications to this device when the pass changes.

## Logging Errors

POST request to `https://webServiceURL/version/log`

### Parameters

*webServiceURL*

The URL to your web service, as specified in the pass.

*version*

The protocol version. Currently, v1.

### Payload

The POST payload is a JSON dictionary, containing a single key and value:

logs (string)

An array of log messages as strings.

### Response

Return HTTP status 200.

### Discussion

This endpoint is intended to help you debug your web service implementation. Log messages contain a description of the error in a human-readable format.

# Document Revision History

This table describes the changes to *Pass Kit Web Service Reference*.

Date	Notes
2012-07-26	New document that describes the web service API used to update passes.





Apple Inc.

© 2012 Apple Inc.

All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Apple Inc.

1 Infinite Loop

Cupertino, CA 95014

408-996-1010

Apple, the Apple logo, and Numbers are trademarks of Apple Inc., registered in the United States and other countries.

iOS is a trademark or registered trademark of Cisco in the U.S. and other countries and is used under license.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.