Pass Kit Package Format Reference

Developer

Contents

Tables

About Pass Files

Important: This is a preliminary document for an API or technology in development. Although this document has been reviewed for technical accuracy, it is not final. This Apple confidential information is for use only by registered members of the applicable Apple Developer program. Apple is supplying this confidential information to help you plan for the adoption of the technologies and programming interfaces described herein. This information is subject to change, and software implemented according to this document should be tested with final operating system software and final documentation. Newer versions of this document may be provided with future seeds of the API or technology.

Companion Guide *Pass Kit Programming Guide*

Passes are a digital representation of information that might otherwise be printed on small pieces of paper or plastic. They let users take an action in the physical world, in the same way as boarding passes, membership cards, and coupons.

At a Glance

This document covers the file format used by the PassKit framework to describe passes.

Understanding The Package Structure

Pass files are stored on disk as a zipped package containing JSON files and other resources.

Relevant Chapters "Package Structure" (page 5)

Understanding The Keys

The `pass.json` file contains a dictionary that contains most of the information about the pass.

Relevant Chapters "Top-Level Keys" (page 6), "Lower-Level Keys" (page 9), "Field Dictionary Keys" (page 11)

Package Structure

Pass files are stored on disk as a zipped package with the `pkpass` file extension.

Localized resources are loaded using the standard bundle localization techniques, which are implemented by the `NSBundle` class. For more details, see *Internationalization Programming Topics*.

The top level of the package contains the following files:

`background.png`
: The image displayed as the background of the front of the pass.

`icon.png`
: The pass's icon. This is displayed in notifications and in emails that have a pass attached, and on the lock screen.

  When it is displayed, the icon gets a shine effect and rounded corners.

`logo.png`
: The image displayed on the front of the pass.

`manifest.json`
: A JSON dictionary. Each key is the path to a file (relative to the top level of the bundle) and the key's value is the SHA-1 hash for that file. Every file in the bundle appears in the manifest, except for the manifest itself and the signature.

`pass.json`
: A JSON dictionary that defines the pass. Its contents are described in detail in "Top-Level Keys" (page 6).

`signature`
: A detached PKCS#7 signature of the `manifest.json` file.

`strip.png`
: The image displayed as a strip behind the primary fields on the front of the pass.

`thumbnail.png`
: An additional image displayed on the front of the pass. For example, on a membership card, the thumbnail could be used to a picture of the cardholder.

Note All of the pass's images are loaded using standard `UIImage` image-loading methods. This means, for example, the file name of high resolution version of the image ends with `@2x.png`.

Top-Level Keys

The top level of the `pass.json` file is a dictionary. The following sections list the required and optional keys used in this dictionary. For each key whose value is a dictionary or an array of dictionaries, there is also a section in "Lower-Level Keys" (page 9) that lists the keys for that dictionary.

Standard Keys

Information that is required for all passes.

Key name	Type	Description
`description`	localizable string	*Required.* Brief description of the pass, used by the iOS accessibility technologies. Don't try to include all of the data on the pass in its description, just include enough detail to distinguish passes of the same type.
`formatVersion`	integer	*Required.* Version of the file format. The value must be 1.
`organizationName`	localizable string	*Required.* Display name of the organization that originated and signed the pass.
`passTypeIdentifier`	string	*Required.* Pass type identifier, as issued by Apple. The value must correspond with your signing certificate.
`serialNumber`	string	*Required.* Serial number that uniquely identifies the pass. No two passes with the same pass type identifier may have the same serial number.
`teamIdentifier`	string	*Required.* Team identifier of the organization that originated and signed the pass, as issued by Apple.

Style Keys

Specifies the pass style.

Provide exactly one key—the key that corresponds with the pass's type. The value of this key is a dictionary containing the keys in "Pass Structure Dictionary Keys" (page 9).

Key name	Type	Description
`boardingPass`	pass-structure dictionary	Information specific to a boarding pass.
`coupon`	pass-structure dictionary	Information specific to a coupon.
`eventTicket`	pass-structure dictionary	Information specific to an event ticket.
`generic`	pass-structure dictionary	Information specific to a generic pass.
`storeCard`	pass-structure dictionary	Information specific to a store card.

Web Service Keys

Information used to update passes using the web service.

Either provide both of these keys or neither of them.

Key name	Type	Description
`authenticationToken`	string	The authentication token to use with the web service. The token must be 16 characters or longer.
`webServiceURL`	string	The URL of a web service that conforms to the API described in *Pass Kit Web Service Reference*. The web service must use the HTTPS protocol; the leading `https://` is included in the value of this key. On devices configured for development, there is UI in Settings to allow HTTP web services.

Relevance Keys

Information about where and when a pass is relevant.

Key name	Type	Description
`locations`	array of location dictionaries	*Optional.* Locations where the pass is relevant. For example, the location of your store. For these dictionaries' keys, see "Location Dictionary Keys" (page 10)
`relevantDate`	ISO 8601 date, as a string	*Optional.* Date and time when the pass becomes relevant. For example, the start time of a movie.

Visual Appearance Keys

Visual styling and appearance of the pass.

Key name	Type	Description
`barcode`	barcode dictionary	*Optional.* Information specific to barcodes. For this dictionary's keys, see "Barcode Dictionary Keys" (page 10)
`backFields`	array of field dictionaries	*Optional.* Information about fields that are displayed on the back of the pass.
`backgroundColor`	color, as a string	*Optional.* Background color of the pass, specified as an CSS-style RGB triple. For example, `rgb(23, 187, 82)`.
`foregroundColor`	color, as a string	*Optional.* Foreground color of the pass, specified as a CSS-style RGB triple. For example, `rgb(100, 10, 110)`.
`labelColor`	color, as a string	*Optional.* Color of the label text, specified as a CSS-style RGB triple. For example, `rgb(255, 255, 255)`. If omitted, the label color is determined automatically.
`logoText`	localizable string	*Optional.* Text displayed next to the logo on the pass.
`suppressStripShine`	Boolean	*Optional.* If `true`, the strip image is displayed without a shine effect. The default value is `false`.

Lower-Level Keys

Keys that are used lower in the hierarchy of the `pass.json` file—for example, in a dictionary that is the value of a top-level key.

Pass Structure Dictionary Keys

Keys that define the structure of the pass.

These keys are used for all pass styles and partition the fields into the various parts of the pass.

Key name	Type	Description
`headerFields`	array of field dictionaries	*Optional.* Fields to be displayed in the header on the front of the pass. Use header fields sparingly; unlike all other fields, they remain visible when a stack of pases are displayed.
`primaryFields`	array of field dictionaries	*Optional.* Fields to be displayed prominently on the front of the pass.
`secondaryFields`	array of field dictionaries	*Optional.* Fields to be displayed on the front of the pass.
`auxiliaryFields`	array of field dictionaries	*Optional.* Additional fields to be displayed on the front of the pass.
`backFields`	array of field dictionaries	*Optional.* Fields to be on the back of the pass.
`transitType`	string	*Required for boarding passes; otherwise not allowed.* Type of transit. Must be one of the following values: `PKTransitTypeAir`, `PKTransitTypeTrain`, `PKTransitTypeBus`, `PKTransitTypeBoat`, `PKTransitTypeGeneric`.

Location Dictionary Keys

Information about a location.

Key name	Type	Description
`altitude`	double	*Optional.* Altitude, in meters, of the location.
`latitude`	double	*Required.* Latitude, in degrees, of the location.
`longitude`	double	*Required.* Longitude, in degrees, of the location.
`relevantText`	string	*Optional.* Text displayed on the lock screen when the pass is currently relevant. For example, a description of the nearby location such as "Store nearby on 1st and Main."

Barcode Dictionary Keys

Information about a pass's barcode.

Key name	Type	Description
`altText`	string	*Optional.* Text displayed near the barcode. For example, a human-readable version of the barcode data in case the barcode doesn't scan.
`format`	string	*Required.* Barcode format. Must be one of the following values: `PKBarcodeFormatQR`, `PKBarcodeFormatPDF417`, `PKBarcodeFormatAztec`.
`message`	string	*Required.* Message or payload to be displayed as a barcode.
`messageEncoding`	IANA character set name, as a string	*Required.* Text encoding that is used to convert the message from the string representation to a data representation to render the barcode. The value is typically `iso-8859-1`, but you may use another encoding that is supported by your barcode scanning infrastructure.

Field Dictionary Keys

Keys that are used at the lowest level of the `pass.json` file, which define an individual field.

Standard Field Dictionary Keys

Information about a field.

These keys are used for all dictionaries that define a field.

Key name	Type	Description
`changeMessage`	localizable format string	*Optional.* Format string for the alert text that is displayed when the pass is updated. The format string may contain the escape `%@`, which is replaced with the field's new value. For example, "Gate changed to %@." If you don't specify a change message, the user isn't notified when the field changes.
`key`	string	*Required.* The key must be unique within the scope of the entire pass. For example, "departure-gate".
`label`	localizable string	*Optional.* Label text for the field.
`textAlignment`	string	*Optional.* Alignment for the field's contents. Must be one of the following values: `PKTextAlignmentLeft`, `PKTextAlignmentCenter`, `PKTextAlignmentRight`, `PKTextAlignmentJustified`, `PKTextAlignmentNatural`
`value`	localizable string, ISO 8601 date as a string, or number	*Required.* Value of the field. For example, `42`.

Date Style Keys

Information about how a date should be displayed in a field.

If any of these keys is present, the value of the field is treated as a date. Either specify both a date style and a time style or neither.

Key name	Type	Description
`dateStyle`	string	Style of date to display. Must be one of the styles listed in Table 4-1 (page 12).
`timeStyle`	string	Style of time to display. Must be one of the styles listed in Table 4-1 (page 12).
`isRelative`	Boolean	If `true`, the label's value is displayed as a relative date; otherwise, it is displayed as an absolute date. The default value is `false`.

The date styles and time styles have the same meaning as the Cocoa formatter styles with corresponding names, as shown in Table 4-1.

Table 4-1 Date and time styles

Date style	Corresponding formatter style
`PKDateStyleNone`	`NSDateFormatterNoStyle`
`PKDateStyleShort`	`NSDateFormatterShortStyle`
`PKDateStyleMedium`	`NSDateFormatterMediumStyle`
`PKDateStyleLong`	`NSDateFormatterLongStyle`
`PKDateStyleFull`	`NSDateFormatterFullStyle`

Number Style Keys

Information about how a number should be displayed in a field.

These keys are optional if the field's value is a number; otherwise they are not allowed.

The number styles have the same meaning as the formatter styles that have corresponding names. For more information, see `NSNumberFormatterStyle`.

Key name	Type	Description
`currencyCode`	string	ISO 4217 currency code for the field's value.

Key name	Type	Description
`numberStyle`	string	Style of number to display. Must be one of the following values: `PKNumberStyleDecimal`, `PKNumberStylePercent`, `PKNumberStyleScientific`, `PKNumberStyleSpellOut`.

Document Revision History

This table describes the changes to *Pass Kit Package Format Reference*.

Date	Notes
2012-08-03	New document that describes the package format used to define a pass.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010