Best Practices and What’s New with In-App Purchases

Session 704

Dana DuBois, App Store Engineering Manager
Ross LeBeau, App Store Engineer
Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Introductory Pricing

Trials

Asking for Ratings and Reviews

Sandbox Testing

Best Practices for Processing Transactions

Techniques for Managing Receipts
Introductory Pricing

Trials

Asking for Ratings and Reviews

Sandbox Testing

Best Practices for Processing Transactions

Techniques for Managing Receipts
Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Introductory Pricing

Ross LeBeau, App Store Engineer
Introductory Pricing

One-time discount for new subscribers
One per subscription
Automatically applied by App Store
Eligibility based on subscription group
Introductory Pricing API

Reflects data from App Store Connect

Used to format and display UI

Used to determine eligibility
Pro Subscription for Forest Explorer
Detailed satellite imagery
1 year for $19.99, then $39.99/year after trial.

INSTALL & SUBSCRIBE
In-App Purchases
// Introductory Pricing API

SKProduct
introductoryPrice: SKProductDiscount?
// Introductory Pricing API

SKProductDiscount
price: NSDecimalNumber
priceLocale: Locale
subscriptionPeriod: SKProductSubscriptionPeriod
  .unit: SKProduct.PeriodUnit
    .day, .week, .month, .year
  .numberOfUnits: Int

numberOfPeriods: Int
// Introductory Pricing API

SKProductDiscount
paymentMode: SKProductDiscount.PaymentMode
  .payAsYouGo
Pay As You Go Example

SKProductDiscount
price = 1.99
subscriptionPeriod
unit = .month
numberOfUnits = 3
numberOfPeriods = 2
paymentMode = .payAsYouGo
// Introductory Pricing API

SKProductDiscount
paymentMode: SKProductDiscount.PaymentMode
   .payAsYouGo
   .payUpFront
Pay Up Front Example

SKProductDiscount
  price = 3.99
  subscriptionPeriod
  numberOfPeriods = 1
  paymentMode = .payUpFront

SKProductSubscriptionPeriod
  unit = .month
  numberOfUnits = 6
// Introductory Pricing API

SKProductDiscount

paymentMode: SKProductDiscount.PaymentMode
  .payAsYouGo
  .payUpFront
  .freeTrial
Introductory Pricing

New in iOS 11.2 and macOS 10.13.2:
SKProduct.subscriptionPeriod: SKProductSubscriptionPeriod

New in iOS 12 and macOS 10.14:
SKProduct.subscriptionGroupIdentifier: String

Introductory Pricing eligibility is based on subscription group

One introductory price per user for each group
More Information

Engineering Subscriptions

Hall 1

Tuesday 3:00PM
Trials

Dana DuBois, App Store Engineering Manager
Free Trials for Non-Subscription Apps
Try before you buy

Non-subscription apps can now offer a free trial

Free app with non-consumable in-app purchase to unlock

Trial Period: Use non-consumable in-app purchase at price tier 0

• Naming convention: 14-day Trial

Clearly inform customers

• Duration of trial
• Cost to unlock full functionality
• Features or content lost when trial ends
Asking for Ratings and Reviews
Asking for Ratings and Reviews
SKStoreReviewController

Introduced in iOS 10.3

Quick way to request a rating/review

Restrictions in place
• Limited requests per device
• Can be disabled by user in Settings
Asking for Ratings and Reviews
SKStoreReviewController

Strategies for when to request review
• Don’t interrupt
• Wait for a positive experience
• Track previous requests
// Asking for Ratings and Reviews with SKStoreReviewController

if shouldPromptUser() {
    SKStoreReviewController.requestReview()
}

func shouldPromptUser() -> Bool {
    // Local business rules
}
“StoreKit has made it so easy for our customers to provide ratings that we are getting over 10x the amount we used to. It gives us confidence that we are delivering the right things to our customers.”

- Zappos

“The Ratings and Reviews API has had enormous positive implications, not only for my App Store visibility, but also for the player insights it affords me as a consequence of a much larger (and arguably more relevant and engaged) sample size.”

- Frosty Pop
Wow, Firewatch is truly one of a kind. I was hooked the moment I started the game, and I still can't stop thinking about some of the story elements. The graphics are crisp, the music is captivating, and the controls are intuitive—do yourself a favor and play this game right now.
Asking for Ratings and Reviews

Deep link to write a review in App Store

Introduced in iOS 10.3

Now supported on macOS

Link to open your app in the App Store
• Presents compose review from app page

User-initiated actions
• Button in Settings
Asking for Ratings and Reviews
Deep link to write a review in App Store

URL is formed using regular product URL with an anchor tag

For creating product URLs visit
https://linkmaker.itunes.apple.com/
More Information

Sandbox Testing
The Sandbox

Environment for testing purchases during development

Based on certificate used to sign your application
The Sandbox

Differences

No charge

Dedicated Sandbox accounts

Backend environment

• https://sandbox.itunes.apple.com/verifyReceipt

Test modes

• SKReceiptRefreshRequest
• SKMutablePayment simulatesAskToBuyInSandbox

Time contraction for subscriptions
## Subscription Timing

### Condensed intervals

<table>
<thead>
<tr>
<th>Actual Duration</th>
<th>Test Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 week</td>
<td>3 minutes</td>
</tr>
<tr>
<td>1 month</td>
<td>5 minutes</td>
</tr>
<tr>
<td>2 months</td>
<td>10 minutes</td>
</tr>
<tr>
<td>3 months</td>
<td>15 minutes</td>
</tr>
<tr>
<td>6 months</td>
<td>30 minutes</td>
</tr>
<tr>
<td>1 year</td>
<td>1 hour</td>
</tr>
</tbody>
</table>

Maximum **five** renewals
The Sandbox
Setting up the test environment

Setup in App Store Connect
• Create test user
• Enter products for sale

Build and sign your app

Buy products
• Sign in with test user when prompted
Apple ID: john.appleseed@icloud.com

Face ID is enabled for all purchases.

AUTOMATIC DOWNLOADS

- **Music**
  - Switch is off
- **Apps**
  - Switch is off
- **Books & Audiobooks**
  - Switch is off
- **Updates**
  - Switch is on
  - Description: Automatically download new purchases (including free) made on other devices.

Use Cellular Data

- Switch is off
  - Description: Use cellular network for automatic downloads.

Video Autoplay

- Switch is on
  - Description: Automatically play app preview videos in the App Store.

In-App Ratings & Reviews

- Switch is on
  - Description: Help developers and other users know what you think by letting apps ask for product feedback.

Offload Unused Apps
Best Practices for Processing Transactions

Ross LeBeau, App Store Engineer
Adding a Transaction Observer

Add as early as possible

Preferably during `application(_:didFinishLaunchingWithOptions:)`

Why?

- Interrupted buy
  - User leaves app
  - User needs to edit billing info
  - App crashes
Adding a Transaction Observer

Why else?

- Subscription renewals
- Promoted In-App Purchases
- In-App Purchase Promo Codes
# Finishing Transactions

<table>
<thead>
<tr>
<th>Transaction State</th>
<th>Action Needed</th>
</tr>
</thead>
<tbody>
<tr>
<td>.purchasing</td>
<td>None. Keep observing the payment queue and wait for the transaction state to change.</td>
</tr>
<tr>
<td>.purchased</td>
<td>Deliver the content to the user, then call finishTransaction.</td>
</tr>
<tr>
<td>.failed</td>
<td>Inspect error and handle as needed, then call finishTransaction.</td>
</tr>
<tr>
<td>.restored</td>
<td>Deliver the content to the user, then call finishTransaction.</td>
</tr>
<tr>
<td>.deferred</td>
<td>None. Keep observing the payment queue.</td>
</tr>
</tbody>
</table>
Ask To Buy

Transaction Deferred

- Approved
  - .purchased
- Declined
  - .failed

24 Hours

Silent Fail
Finishing Transactions

Finish downloading hosted content first
  • Downloads will cancel and become unavailable

Verify the receipt before finishing the transaction
  • Consumables will not appear after finished
Techniques for Managing Receipts

Dana DuBois, App Store Engineering Manager
The Receipt

- Trusted record of App and In-App Purchases
- Stored on device
- Issued by the App Store
- Signed and verifiable
- For your app, on that device only
Receipt Validation

On-device validation
• Unlock features and content within the app

Server-to-server validation
• Online validation through a request to the App Store
• Unlock features/subscription state on your server

Engineering Subscriptions  Hall 1  Tuesday 3:00PM
Receipt Validation

Server-to-server validation

- Use a trusted server to communicate with the App Store
- Do NOT post the receipt directly to the App Store to be validated
On-Device Receipt Validation

The basics

Stored in the app bundle
• API to get the path

Single file
• Purchase data
• Signature to check authenticity
On-Device Receipt Validation

Standards

Signing
• PKCS#7 Cryptographic Container

Data Encoding
• ASN.1

Options for verifying and reading
• OpenSSL, asn1c, and more
• Create your own
On-Device Receipt Validation
The basics

Locate the receipt using Bundle API

```swift
// Locate the file
guard let url = Bundle.main.appStoreReceiptURL else {
    // Handle failure
    return
}
// Read the contents
let receipt = Data(contentsOf: url)
```
On-Device Receipt Validation
Tips for using OpenSSL

Build your own static library (.a file)
• Not a dynamic library

Include Apple Root CA Certificate
• Available online
• If bundled in app, watch out for expiry

Documentation online
On-Device Receipt Validation

Downloading pre-built solutions
On-Device Receipt Validation

Downloading pre-built solutions

Convenience comes at a price

• Reusing code brings with it bugs and vulnerabilities
• Single exploit affects many
On-Device Receipt Validation

Downloading pre-built solutions

Convenience comes at a price

• Reusing code brings with it bugs and vulnerabilities
• Single exploit affects many

It’s your revenue stream

• Make decisions that suit your product
• Know and own the risks
On-Device Receipt Validation
Certificate verification

- Do not check the expiry date of the certificate relative to current date

- Compare expiry date to purchase date of the transaction
On-Device Receipt Validation

Receipt payload

Series of attributes
• Type
• Value
• (Version)
On-Device Receipt Validation

Verify application

Check the bundle identifier

Check the bundle version

Use hardcoded values
  • Not Info.plist values
On-Device Receipt Validation

Verify application

Attribute 5 is a SHA-1 hash of three key values
- Bundle identifier
- Device identifier
- Opaque value (Attribute 4)

Unique to your app on this device

Create hash using hardcoded values, compare
On-Device Receipt Validation

Receipt refresh

If the receipt doesn’t exist or is invalid, refresh the receipt using StoreKit

Receipt refresh will require network

Store sign-in will be required

Avoid continuous loop of validate-and-refresh

```swift
let request = SKReceiptRefreshRequest()
request.delegate = self
request.start()
```
On-Device Receipt Validation

Receipt refresh

If the receipt doesn’t exist or is invalid, refresh the receipt using StoreKit

Receipt refresh will require network

Store sign-in will be required

Avoid continuous loop of validate-and-refresh

// macOS only

exit(173)
On-Device Receipt Validation

Free Trials

In-App Purchase Receipt

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>1702</td>
<td>Product Identifier</td>
</tr>
<tr>
<td>1704</td>
<td>Purchase Date</td>
</tr>
</tbody>
</table>

Receipt

Purchase Information

Attribute

<table>
<thead>
<tr>
<th>Type</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Bundle Identifier</td>
</tr>
</tbody>
</table>

In-App Purchase

Attribute

<table>
<thead>
<tr>
<th>Type</th>
<th>In-App Purchase</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td></td>
</tr>
</tbody>
</table>

Type 1702 Product Identifier
Type 1704 Purchase Date
On-Device Receipt Validation

Free Trials

Iterate over In-App Purchases

If any has “Full Unlock” Product Identifier, unlock the app

If any has “Free Trial” Product Identifier, compare current date to Purchase Date

If neither are present, display UI to begin “Free Trial”
Changing Business Models

When moving from
• Paid upfront to subscription
• Paid upfront to free trial

Maintain functionality that users have already paid for

Original application version in receipt
Summary

Introductory Pricing

Trials

Asking for Ratings and Reviews

Sandbox Testing

Best Practices for Processing Transactions

Techniques for Managing Receipts
Summary

Introductory Pricing

Trials

Asking for Ratings and Reviews

Sandbox Testing

Best Practices for Processing Transactions

Techniques for Managing Receipts
Summary

Introductory Pricing

Trials

Asking for Ratings and Reviews

Sandbox Testing

Best Practices for Processing Transactions

Techniques for Managing Receipts
Summary

Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Summary

Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Summary

Introductory Pricing
Trials
Asking for Ratings and Reviews
Sandbox Testing
Best Practices for Processing Transactions
Techniques for Managing Receipts
Summary

Introductory Pricing

Trials

Asking for Ratings and Reviews

Sandbox Testing

Best Practices for Processing Transactions

Techniques for Managing Receipts
## More Information

https://developer.apple.com/session704

<table>
<thead>
<tr>
<th>Event</th>
<th>Location</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering Subscriptions</td>
<td>Hall 1</td>
<td>Tuesday 3:00PM</td>
</tr>
<tr>
<td>StoreKit / Subscriptions Lab</td>
<td>Technology Lab 12</td>
<td>Tuesday 4:00PM</td>
</tr>
<tr>
<td>StoreKit / Subscriptions Lab</td>
<td>Technology Lab 12</td>
<td>Thursday 9:00AM</td>
</tr>
</tbody>
</table>