Subscription Offers Best Practices

Ross LeBeau, App Store Engineer
Michael Gargas, Technical Advocate, App Store Commerce
Overview
Overview

Discounted price for specific duration
Overview

Discounted price for specific duration

Powerful tool for mitigating churn
Overview

Discounted price for specific duration

 Powerful tool for mitigating churn

 You choose when to offer to customers
Overview

Discounted price for specific duration

Powerful tool for mitigating churn

You choose when to offer to customers

Customers may redeem as many offers as you allow
Overview

Discounted price for specific duration

Powerful tool for mitigating churn

You choose when to offer to customers

Customers may redeem as many offers as you allow

For existing or previously subscribed customers
Setup

Generating a signature
Interacting with the StoreKit API
Determining customer eligibility
Distributing offers
Using offers to reduce churn
Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn
Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn
Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn
Setup
Generating a signature
Interacting with the StoreKit API
Determining customer eligibility
Distributing offers
Using offers to reduce churn
Setup

Generating a signature

Interacting with the StoreKit API

Determining customer eligibility

Distributing offers

Using offers to reduce churn
Setup
# In-App Purchase

## One Year All Access

### Approved

<table>
<thead>
<tr>
<th>Reference Name</th>
<th>Apple ID</th>
<th>Product ID</th>
<th>Group Reference Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>One Year All Access</td>
<td>48103821</td>
<td>one-year-access_1350231</td>
<td>Forest Explorer</td>
</tr>
</tbody>
</table>

### Subscription Prices

- **Plan Subscription Price Change**
- **Create Introductory Offer**
- **Create Promotional Offer**

### App Store Information

Provide a display name and description for your in-app purchases, and we’ll show this on the App Store.

### Localizations

- **English (US)**
  - **Subscription Display Name**: One Year All Access
  - **Descriptions**: Gain one year access to the complete app!
Create a Promotional Offer

Reference and product code names can be used only once, even if the promotional offer is deleted.

Promotional Offer Reference Name
One Year All Access Promo

Promotional Offer Product Code
ALLACCESS
Type of Promotional Offer

Select a paid or free promotional offer. Subscribers will automatically be charged the full subscription price when the promotional offer is over unless they cancel their subscription. Learn More.

- Pay as you go
- Pay up front
- Free

Duration

Pay as you go for 12 months.

Price

Choose a price, and we’ll automatically calculate the prices for all 155 territories based on the most recent foreign exchange rates. You can add prices for individual territories in the next step.

Currency: USD - US Dollar
Price: $4.99 for the first 6 months
Generating a subscription key allows Apple to authenticate and validate a subscription request. Keys can't be modified once created. You can have a maximum of 10 active keys at a time. Learn More

### Active (1)

<table>
<thead>
<tr>
<th>Name</th>
<th>Generated By</th>
<th>Key ID</th>
<th>Downloaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forest Explorer Developer Key</td>
<td>Cindy Cheang</td>
<td>E9HJXAZV2</td>
<td>Mar 26, 2019</td>
</tr>
</tbody>
</table>

### Revoked (2)

Revoked keys are displayed for 30 days.

<table>
<thead>
<tr>
<th>Name</th>
<th>Revoked By</th>
<th>Key ID</th>
<th>Revoked On</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ocean Jewel Developer Key</td>
<td>Stephanie Vital</td>
<td>23HYFHVTH</td>
<td>Feb 14, 2019</td>
</tr>
<tr>
<td>Global Developer Key</td>
<td>Sarah Mitas</td>
<td>SJFJNTA:A.B</td>
<td>Feb 12, 2019</td>
</tr>
</tbody>
</table>
### Users and Access

#### Active (1)

<table>
<thead>
<tr>
<th>NAME</th>
<th>GENERATED BY</th>
<th>KEY ID</th>
<th>Downloaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forest Explorer Developer Key</td>
<td>Cindy Chiang</td>
<td>5QKJMN3ZC6Q</td>
<td>Mar 29, 2019</td>
</tr>
</tbody>
</table>

#### Revoked (2)

- Revoked keys are displayed for 30 days.

<table>
<thead>
<tr>
<th>NAME</th>
<th>Revoked On</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nature Lab Key 2019</td>
<td>Feb 14, 2019</td>
</tr>
<tr>
<td>Global Developer Key</td>
<td>Feb 15, 2019</td>
</tr>
</tbody>
</table>

---

**Generate Subscription Key**

- **Name**: Nature Lab Key 2019

  - **Cancel**
  - **Generate**
### Users and Access

**Key Type**
- App Store Connect API
- Subscriptions

Generating a subscription key allows Apple to authenticate and validate a subscription request. Keys can't be modified once created. You can have a maximum of 10 active keys at a time. [Learn More]

#### Active (2)

<table>
<thead>
<tr>
<th>NAME</th>
<th>GENERATED BY</th>
<th>KEY ID</th>
<th>Downloaded</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nature Lab Key 2019</td>
<td>John Applied</td>
<td>2P2D96K4OK Copy Key ID</td>
<td></td>
</tr>
<tr>
<td>Forest Explorer Developer Key</td>
<td>Cindy Cheung</td>
<td>D9HJNTGAQ</td>
<td>Mar 25, 2019</td>
</tr>
</tbody>
</table>

#### Revoked (2)

Revoked keys are displayed for 30 days.

<table>
<thead>
<tr>
<th>NAME</th>
<th>REVOKED BY</th>
<th>KEY ID</th>
<th>REVOKED ON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oculus Journal Developer Key</td>
<td>Stephanie Vitali</td>
<td>20YJHJHEFH</td>
<td>Feb 14, 2019</td>
</tr>
<tr>
<td>Global Developer Key</td>
<td>Sarah Mios</td>
<td>5JSJFNATJLB</td>
<td>Feb 12, 2019</td>
</tr>
</tbody>
</table>
Subscription Offers Keys
Subscription Offers Keys

Valid across entire developer account

• All apps
• All offers

Multiple keys can be active at once
Generating a Signature
Cryptographic Signature
Cryptographic Signature

Ensures only authorized users can redeem offers
Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature
Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

Asymmetric cryptography
Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

Asymmetric cryptography

• Private key used to sign payload
Cryptographic Signature

Ensures only authorized users can redeem offers

Payload and signature

Asymmetric cryptography

• Private key used to sign payload
• Public key used to validate signature
Getting the payload data

router.get('/offer', function(req, res) {
    const appBundleID = req.body.appBundleID;
    const productIdentifier = req.body.productIdentifier;
    const subscriptionOfferID = req.body.offerID;
    const applicationUsername = req.body.applicationUsername;

    const nonce = uuidv4();

    const currentDate = new Date();
    const timestamp = currentDate.getTime();

    // ...
});
router.get('/offer', function(req, res) {
  const appBundleID = req.body.appBundleID;
  const productIdentifier = req.body.productIdentifier;
  const subscriptionOfferID = req.body.offerID;
  const applicationUsername = req.body.applicationUsername;

  const nonce = uuidv4();

  const currentDate = new Date();
  const timestamp = currentDate.getTime();

  // ...
});
//Getting the payload data

router.get('/offer', function(req, res) {
    const appBundleID = req.body.appBundleID;
    const productIdentifier = req.body.productIdentifier;
    const subscriptionOfferID = req.body.offerID;
    const applicationUsername = req.body.applicationUsername;

    const nonce = uuidv4();

    const currentDate = new Date();
    const timestamp = currentDate.getTime();

    // ...
});
//Getting the payload data

router.get('/offer', function(req, res) {
  const appBundleID = req.body.appBundleID;
  const productIdentifier = req.body.productIdentifier;
  const subscriptionOfferID = req.body.offerID;
  const applicationUsername = req.body.applicationUsername;

  const nonce = uuidv4();

  const currentDate = new Date();
  const timestamp = currentDate.getTime();

  // ...
//Getting the key ID

router.get('/offer', function(req, res) {
  // ...

  const keyID = getKeyID();

  // ...
});

function getKeyID() {
  // Implement key selection logic here
  return process.env.SUBSCRIPTION_OFFERS_KEY_ID;
}
// Getting the key ID

router.get('/offer', function(req, res) {
    // ...

    const keyID = getKeyID();

    // ...

});

function getKeyID() {
    // Implement key selection logic here
    return process.env.SUBSCRIPTION_OFFERS_KEY_ID;
}
// Getting the key ID

router.get('/offer', function(req, res) {
    // ...

    const keyID = getKeyID();

    // ...
});

function getKeyID() {
    // Implement key selection logic here

    return process.env.SUBSCRIPTION_OFFERS_KEY_ID;
}
//Creating the payload string

router.get('/offer', function(req, res) {
    // ...

    const payload = appBundleID + '\u2063' +
                    keyID + '\u2063' +
                    productIdentifier + '\u2063' +
                    subscriptionOfferID + '\u2063' +
                    applicationUsername + '\u2063' +
                    nonce + '\u2063' +
                    timestamp;

});
// Getting the private key

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    // ...

});

function getKeyStringForID(keyID) {
    if (keyID === process.env.SUBSCRIPTION_OFFERS_KEY_ID) {
        return process.env.SUBSCRIPTION_OFFERS_PRIVATE_KEY;
    } else {
        throw "Key ID not recognized";
    }
}

// Getting the private key

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    // ...
});

function getKeyStringForID(keyID) {
    if (keyID === process.env.SUBSCRIPTION_OFFERS_KEY_ID) {
        return process.env.SUBSCRIPTION_OFFERS_PRIVATE_KEY;
    } else {
        throw "Key ID not recognized";
    }
}
// Getting the private key

router.get('/offer', function(req, res) {
    // ...
    
    const keyString = getKeyStringForID(keyID);
    // ...
});

function getKeyStringForID(keyID) {
    if (keyID === process.env.SUBSCRIPTION_OFFERS_KEY_ID) {
        return process.env.SUBSCRIPTION_OFFERS_PRIVATE_KEY;
    } else {
        throw "Key ID not recognized";
    }
}
// Generating the signature

router.get('/offer', function(req, res) {
   // ...

   const keyString = getKeyStringForID(keyID);

   const key = new ECKey(keyString, 'pem');

   const cryptoSign = key.createSign('SHA256');

   cryptoSign.update(payload);

   const signature = cryptoSign.sign('base64');
});
// Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
// Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');
});
//Generating the signature

router.get('/offer', function(req, res) {
    // ...

    const keyString = getKeyStringForID(keyID);

    const key = new ECKey(keyString, 'pem');

    const cryptoSign = key.createSign('SHA256');

    cryptoSign.update(payload);

    const signature = cryptoSign.sign('base64');

});
// Verifying the signature and sending the response

router.get('/offer', function(req, res) {
    // ...

    const verificationResult = key.createVerify('SHA256').update(payload).verify(signature, 'base64');
    console.log("Verification result: " + verificationResult)

    res.setHeader('Content-Type', 'application/json');
    res.json({
        'keyID': keyID,  
        'nonce': nonce,  
        'timestamp': timestamp,  
        'signature': signature
    });
});
Verifying the signature and sending the response

```javascript
const verificationResult = key.createVerify('SHA256').update(payload).verify(signature, 'base64');
console.log("Verification result: " + verificationResult)

res.setHeader('Content-Type', 'application/json');
res.json({
  'keyID': keyID,
  'nonce': nonce,
  'timestamp': timestamp,
  'signature': signature
});
```
// Verifying the signature and sending the response

const verificationResult = key.createVerify('SHA256').update(payload).verify(signature, 'base64);
console.log("Verification result: " + verificationResult)

res.setHeader('Content-Type', 'application/json');
res.json({
  'keyID': keyID,
  'nonce': nonce,
  'timestamp': timestamp,
  'signature': signature
});
Interacting with the StoreKit API
Interacting with the StoreKit API
Interacting with the StoreKit API

Get offer details from the App Store
Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server
Interacting with the StoreKit API

Get offer details from the App Store
Send signature request to server
• Send request when customer is ready to purchase
Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server

• Send request when customer is ready to purchase
• E.g. right before displaying your offer or store UI
Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server
- Send request when customer is ready to purchase
- E.g. right before displaying your offer or store UI
- This ensures recent timestamp and up-to-date business logic
Interacting with the StoreKit API

Get offer details from the App Store

Send signature request to server
• Send request when customer is ready to purchase
• E.g. right before displaying your offer or store UI
• This ensures recent timestamp and up-to-date business logic

Handle signature response from server
// Getting Offer Details

open class SKProduct: NSObject {
    open var discounts: [SKProductDiscount] { get }
}

/Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}

open class SKProductDiscount : NSObject {
    open var price: NSDecimalNumber { get }
    open var priceLocale: Locale { get }
    open var identifier: String? { get }
    open var subscriptionPeriod: SKProductSubscriptionPeriod { get }
    open var numberOfPeriods: Int { get }
    open var paymentMode: SKProductDiscount.PaymentMode { get }
    open var type: SKProductDiscount.Type { get }
}
/ Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}

class SKProductDiscount : NSObject {
    open var price: NSDecimalNumber { get }
    open var priceLocale: Locale { get }
    open var identifier: String? { get }
    open var subscriptionPeriod: SKProductSubscriptionPeriod { get }
    open var numberOfPeriods: Int { get }
    open var paymentMode: SKProductDiscount.PaymentMode { get }
    open var type: SKProductDiscount.Type { get }
}
// Getting Offer Details

open class SKProduct : NSObject {
    open var discounts: [SKProductDiscount] { get }
}

open class SKProductDiscount : NSObject {
    open var price: NSDecimalNumber { get }
    open var priceLocale: Locale { get }
    open var identifier: String? { get }
    open var subscriptionPeriod: SKProductSubscriptionPeriod { get }
    open var numberOfPeriods: Int { get }
    open var paymentMode: SKProductDiscount.PaymentMode { get }
    open var type: SKProductDiscount.Type { get }
}
// Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {
    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in
        // Handle response
    })
}
// Fetching offer details

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        // Handle response
    })

}
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        // Handle response

    })
}

// Fetching offer details
/*Fetching offer details*/

```swift
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {
    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in
        // Handle response
    })
}
```
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        // Handle response
        }
    )
}
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        // Handle response
    })
}

// Fetching offer details
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        // Handle response
        })

}
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in
        // Fetching offer details
        // Handle response
    })
}
public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        // Handle response

    })
}

public func prepareOffer(usernameHash: String, productIdentifier: String, offerIdentifier: String, completion: (SKPaymentDiscount) -> Void) {

    YourServer.fetchOfferDetails(username: usernameHash, productIdentifier: productIdentifier, offerIdentifier: offerIdentifier, completion: { (nonce: UUID, timestamp: NSNumber, keyIdentifier: String, signature: String) in

        let discountOffer = SKPaymentDiscount(identifier: offerIdentifier, keyIdentifier: keyIdentifier, nonce: nonce, signature: signature, timestamp: timestamp)

        completion(discountOffer)

    })
}
/Sending a payment with a subscription offer to the App Store

```swift
public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```
// Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
// Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
// Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
/Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
/Sending a payment with a subscription offer to the App Store

```swift
public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
```
public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
// Sending a payment with a subscription offer to the App Store

public func buyProduct(product: SKProduct, forApplicationUsername usernameHash: String, withOffer offer: SKPaymentDiscount) {

    let payment = SKMutablePayment(product: product)

    payment.applicationUsername = usernameHash

    payment.paymentDiscount = offer

    SKPaymentQueue.default().add(payment)
}
New SKError Codes
## New SKError Codes

<table>
<thead>
<tr>
<th>SKError.Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.invalidOfferIdentifier</td>
<td>Offer identifier cannot be found, or offer is inactive</td>
</tr>
</tbody>
</table>
## New SKError Codes

<table>
<thead>
<tr>
<th>SKError.Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>.invalidOfferIdentifier</td>
<td>Offer identifier cannot be found, or offer is inactive</td>
</tr>
<tr>
<td>.invalidOfferPrice</td>
<td>Pay-as-you-go offer price is higher than base subscription price</td>
</tr>
</tbody>
</table>
## New SKError Codes

<table>
<thead>
<tr>
<th>SKError.Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>invalidOfferIdentifier</td>
<td>Offer identifier cannot be found, or offer is inactive</td>
</tr>
<tr>
<td>invalidOfferPrice</td>
<td>Pay-as-you-go offer price is higher than base subscription price</td>
</tr>
<tr>
<td>invalidSignature</td>
<td>Signature in SKPaymentDiscount did not successfully validate against public key</td>
</tr>
</tbody>
</table>
### New SKError Codes

<table>
<thead>
<tr>
<th>SKError.Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>invalidOfferIdentifier</code></td>
<td>Offer identifier cannot be found, or offer is inactive</td>
</tr>
<tr>
<td><code>invalidOfferPrice</code></td>
<td>Pay-as-you-go offer price is higher than base subscription price</td>
</tr>
<tr>
<td><code>invalidSignature</code></td>
<td>Signature in SKPaymentDiscount did not successfully validate against public key</td>
</tr>
<tr>
<td><code>missingOfferParams</code></td>
<td>One or more payload data items is missing (for example, empty applicationUsername)</td>
</tr>
</tbody>
</table>
References

Generating a Subscription Offer Using Node.js
Implementing Subscription Offers in Your App
Generating a Signature for Subscription Offers
Determining Customer Eligibility
Determining Eligibility
Determining Eligibility

Must have previously subscribed
Determining Eligibility

Must have previously subscribed
• Any auto-renewable subscription
Determining Eligibility

Must have previously subscribed

• Any auto-renewable subscription
• Any subscription group
Determining Eligibility

Must have previously subscribed
- Any auto-renewable subscription
- Any subscription group

Includes both current and lapsed subscribers
Determining Eligibility

Must have previously subscribed

• Any auto-renewable subscription
• Any subscription group

Includes both current and lapsed subscribers

Eligible on first subscription period
Determining Eligibility

Must have previously subscribed
• Any auto-renewable subscription
• Any subscription group

Includes both current and lapsed subscribers

Eligible on first subscription period
• Includes introductory offers
Determining Eligibility

Must have previously subscribed
• Any auto-renewable subscription
• Any subscription group

Includes both current and lapsed subscribers

Eligible on first subscription period
• Includes introductory offers
• Includes free trials
Creating the SKPaymentDiscount

{
    "status": 0,
    "receipt": {
        "in_app": [
            {
                "quantity": "1",
                "product_id": "com.example.apple.non_consumable_1",
            },
            {
                "quantity": "1",
                "product_id": "com.example.apple.non_consumable_2",
            },
            {
                "quantity": "1",
                "product_id": "com.example.apple.auto_renewable",
            }
        ]
    }
}
//Creating the SKPaymentDiscount

{
"status": 0,
"receipt": {
  "in_app": [
    {
      "quantity": "1",
      "product_id": "com.example.apple.non_consumable_1",
    },
    {
      "quantity": "1",
      "product_id": "com.example.apple.non_consumable_2",
    },
    {
      "quantity": "1",
      "product_id": "com.example.apple.auto_renewable",
    }
  ]
}
}
//Creating the SKPaymentDiscount

{
    "status": 0,
    "receipt": {
        "in_app": [
            {
                "quantity": "1",
                "product_id": "com.example.apple.non_consumable_1",
            },
            {
                "quantity": "1",
                "product_id": "com.example.apple.non_consumable_2",
            },
            {
                "quantity": "1",
                "product_id": "com.example.apple.auto_renewable",
            }
        ]
    }
}
//Creating the SKPaymentDiscount
{
    "status": 0,
    "receipt": {
        "in_app": [
            {
                "quantity": "1",
                "product_id": "com.example.apple.non_consumable_1",
            },
            {
                "quantity": "1",
                "product_id": "com.example.apple.non_consumable_2",
            },
            {
                "quantity": "1",
                "product_id": "com.example.apple.auto_renewable",
            }
        ]
    }
}
Eligibility, Distribution, and Reducing Churn Using Offers

Michael Gargas, Technical Advocate, App Store Commerce
Has a customer churned?

Why did a customer cancel?

Is my price increase reasonable?

When do the majority of customers churn?

Have they subscribed in the past?

Have there been multiple billing issues?

How many consecutive renewal periods?

Was a customer refunded?

Is my customer engaged?

Has a customer churned?

Did the subscriber upgrade or downgrade?

Have they subscribed in the past?
Determining Customer Eligibility
Receipt data
Determining Customer Eligibility
Receipt data

App Receipt
• in_app array
• latest_receipt_info
Determining Customer Eligibility

Receipt data

App Receipt
- in_app array
- latest_receipt_info

Server-to-server notifications
Determining Customer Eligibility
Receipt data
Determining Customer Eligibility
Receipt data

Subscription status
Determining Customer Eligibility

Receipt data

Subscription status

Unique identifiers
Determining Customer Eligibility

Receipt data

Subscription status

Unique identifiers

Subscriber dates
Determining Customer Eligibility
Receipt data

Subscription status
Unique identifiers
Subscriber dates
Subscriber intent
Determining Customer Eligibility
Storing user states

```json
{ receipt: {
  in_app: [{
    transaction_id: "1234567890",
    original_transaction_id: "1133557799",
    expires_date: "2018-07-08...", ...
  }, {
    transaction_id: "223445566",
    original_transaction_id: "1133557799",
    expires_date: "2018-08-08...", ...
  }]
}
```
Determining Customer Eligibility

Storing user states

```json
{
  receipt: {
    in_app: [{
      transaction_id: "1234567890",
      original_transaction_id: "1133557799",
      expires_date: "2018-07-08...", ...
    }, {
      transaction_id: "2233445566",
      original_transaction_id: "1133557799",
      expires_date: "2018-08-08...", ...
    }]
  }
}
```

Your Server

/processTransaction?userId=90000001

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>latestExpiresDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2018-07-08...</td>
</tr>
</tbody>
</table>

isUserSubscribed()
Determining Customer Eligibility

Storing subscriber status fields
Determining Customer Eligibility
Storing subscriber status fields

Save decoded JSON from /verifyReceipt

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionIdentifier</th>
<th>latestReceiptData</th>
<th>latestExpiresDate</th>
<th>latestDecodedReceipt</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1234567890</td>
<td>d24Fs...kJ87dDGe3=</td>
<td>2019-08-08...</td>
<td>{ ... auto_renew_status: 1 ...}</td>
</tr>
</tbody>
</table>
## Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

Parse out specific JSON fields

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionIdentifier</th>
<th>latestReceiptData</th>
<th>latestExpiresDate</th>
<th>autoRenewStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1234567890</td>
<td>d24Fs...kJ87dDGe3=</td>
<td>2019-08-08...</td>
<td>1</td>
</tr>
</tbody>
</table>
Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

Parse out specific JSON fields

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionIdentifier</th>
<th>latestReceiptData</th>
<th>latestExpiresDate</th>
<th>autoRenewStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1234567890</td>
<td>d24Fs...kJ87dDGe3=</td>
<td>2019-08-08...</td>
<td>1</td>
</tr>
</tbody>
</table>
Determining Customer Eligibility

Storing subscriber status fields

Save decoded JSON from `/verifyReceipt`

Parse out specific JSON fields

Save status changes from server-to-server notifications

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionIdentifier</th>
<th>latestReceiptData</th>
<th>latestExpiresDate</th>
<th>autoRenewStatus</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1234567890</td>
<td>d24Fs...kJ87dDGe3=</td>
<td>2019-08-08...</td>
<td>1</td>
</tr>
</tbody>
</table>
Determining Customer Eligibility
Developer data
Determining Customer Eligibility
Developer data

Developers have data Apple does not

Marry receipt data with customer subscription activity

Curate offers for specific cohorts of users

Monitor how users respond to offers using:
- Conversion
- Engagement
- Retention
- Churn
<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>latestExpiresDate</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>userId</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
</tr>
<tr>
<td>-----------</td>
<td>-----------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
</tr>
<tr>
<td>userId</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
</tr>
<tr>
<td>userid</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
</tr>
<tr>
<td>userId</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
</tr>
<tr>
<td>----------</td>
<td>-----------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
</tr>
<tr>
<td>userId</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
</tr>
<tr>
<td>----------</td>
<td>------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
</tr>
<tr>
<td>userId</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------</td>
<td>-------------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
</tr>
</tbody>
</table>
Determining Customer Eligibility

Customer cohorts
## Determining Customer Eligibility

### Customer cohorts

| userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | retentionOffer | upgradeOffer |
|--------|----------------|-----------------|--------------------|-------------------|-----------------|--------------|---------------|--------------|--------------|
### Determining Customer Eligibility

#### Customer cohorts

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>retentionOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>90000003</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
### Determining Customer Eligibility

#### Customer cohorts

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>retentionOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>90000003</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
## Determining Customer Eligibility

### Customer cohorts

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>retentionOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>90000003</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
## Determining Customer Eligibility

### Customer cohorts

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>retentionOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>90000003</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Generating a signature
Interacting with the StoreKit API
Determining customer eligibility
Distributing offers
Using offers to reduce churn
Distributing Offers

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>retentionOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>90000003</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>userId</td>
<td>renewalPeriods</td>
<td>autoRenewStatus</td>
<td>billingRetryStatus</td>
<td>billingIssueCount</td>
<td>contentConsumed</td>
<td>winbackOffer</td>
<td>retentionOffer</td>
<td>upgradeOffer</td>
</tr>
<tr>
<td>------------</td>
<td>----------------</td>
<td>-----------------</td>
<td>--------------------</td>
<td>-------------------</td>
<td>-----------------</td>
<td>--------------</td>
<td>----------------</td>
<td>--------------</td>
</tr>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>90000003</td>
<td>10</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Distributing Offers
Presentation

In-App
• Messaging
• Notifications
Distributing Offers
Presentation
Distributing Offers
Presentation

Landing point for offer consumption

Think about the call to action

Comply with App Review Guidelines for subscriptions
Distributing Offers

Presenting

What's New in Universal Links

WWDC App
Distributing Offers

Presenting

External channels

• Paid advertising
• Email marketing

What's New in Universal Links

WWDC App
Distributing Offers

Presenting

External channels
• Paid advertising
• Email marketing

Universal links
• Drive users back into the app from external channels

What's New in Universal Links
Generating a signature
Interacting with the StoreKit API
Determining customer eligibility
Distributing offers
Using offers to reduce churn
Using Offers to Reduce Churn

Voluntary churn
Using Offers to Reduce Churn

Voluntary churn

Time

Subscription Period

Transaction

Subscription Period

Transaction

Subscription Period

Cancellation

Transaction
Using Offers to Reduce Churn

Voluntary churn
Using Offers to Reduce Churn

Voluntary churn

Analyze customer subscription behavior
Using Offers to Reduce Churn

Voluntary churn

Analyze customer subscription behavior

Identify customer cohorts
Using Offers to Reduce Churn

Voluntary churn

Analyze customer subscription behavior

Identify customer cohorts

Distribute relevant subscription offers
Using Offers to Reduce Churn

Offer strategies
Using Offers to Reduce Churn

Offer strategies

- Win Backs
- Retention Marketing
- Save Offers
- Upgrade/Downgrade
- Customer Service
- Loyalty
Using Offers to Reduce Churn

Winback

Mar. 26

in_app: [{
  "purchase_date": "2019-03-26...",
  "expires_date": "2019-04-26...",
}]

Apr. 26
Using Offers to Reduce Churn

Winback

in_app: [{
  "purchase_date": "2019-03-26...",
  "expires_date": "2019-04-26...",
  "cancellation_date": "2019-04-04...",
  "cancellation_reason": "1", ...
}]

Mar. 26

Apr. 4

Apr. 26

Cancelled
Using Offers to Reduce Churn

Winback

Cancelled

Mar. 26 - Apr. 26

in_app: [{
    "purchase_date": "2019-03-26...",
    "expires_date": "2019-04-26...",
    "cancellation_date": "2019-04-04...",
    "cancellation_reason": "1", ...
}]}
Using Offers to Reduce Churn
Server-to-server notification

{ notification_type: "CANCEL", ... }

App Store

Your Server

Website
Using Offers to Reduce Churn
Server-to-server notification

{ notification_type: "CANCEL", ... }

App Store

Your Server

Website

Please see alternative subscription offers

Close
Using Offers to Reduce Churn
Winback

```json
pending_renewal_info:
[
  {
    "original_transaction_id": "1133557799",
    "product_id": "plus_subscription_999_6m",
    "auto_renew_status": "0",
    "cancellation_date": "2019-04-04...",
    "cancellation_reason": "1",
    "expiration_intent": "1",
  }
]
```

```
Your Server

/processTransaction?userId=90000001

+----------+------------+--------------+
| userId   | originalTransactionId | expirationIntent |
|----------+-----------------------+------------------|
| 90000001 | 1133557799            | 1                |
+----------+-----------------------+------------------+
```
Using Offers to Reduce Churn
Winback

pending_renewal_info: [{
    "original_transaction_id": "1133557799",
    "product_id": "plus_subscription_999_6m",
    "auto_renew_status": "0",
    "cancellation_date": "2019-04-04...",
    "cancellation_reason": "1",
    "expiration_intent": "1", ...
}]

/过程Transaction?userId=90000001

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>expirationIntent</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>1</td>
</tr>
</tbody>
</table>
# Using Offers to Reduce Churn

## Winback

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Using Offers to Reduce Churn

Winback

Display Subscription Offer

Mar. 26 - Apr. 4 - Apr. 26

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000003</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Explore with us again.
We've noticed you won't be staying with us next month,
Subscribe now to get another month free.

Get 1 month for free.
Then $4.99/month

Subscribe now
Here’s a free month just for you.

Try another 30 days.
Then $4.99 per month.

Reactivate for Free

Subscription automatically renews unless auto-renew is turned off at least 24-hours before the end of the current period. Payment will be charged to iTunes Account at confirmation of purchase. Account may be charged for renewal within 24-hours to the end of the current period and unless cancelled within 24-hours. Cancellation is automatic thereafter. New users only. Offers may not be available in all countries. Prices subject to the then-current exchange rate. Additional terms apply.
Using Offers to Reduce Churn
Retention

Mar. 26 - Apr. 26

in_app: [{
    "purchase_date": "2019-03-26...",
    "expires_date": "2019-04-26...",...
}]


Using Offers to Reduce Churn

Retention

Mar. 30

Disabled Renewal

Mar. 26

Mar. 30

Apr. 26

{in_app: [{
   "purchase_date": "2019-03-26...",
   "expires_date": "2019-04-26...",
}]}
Using Offers to Reduce Churn

Retention

Disabled Renewal

Mar. 26  Mar. 30  Apr. 26
Using Offers to Reduce Churn

Retention

Disabled Renewal

Mar. 26  Mar. 30  Apr. 26
Using Offers to Reduce Churn

Retention

Disabled Renewal

Mar. 26

Mar. 30

Apr. 26

in_app: [
    {
        "purchase_date": "2019-03-26...",
        "expires_date": "2019-04-26..."
        "auto_renew_status": "0",
    }
]
Using Offers to Reduce Churn

Server-to-server notification

```
{ notification_type: "DID_CHANGE_RENEWAL_STATUS", ... }
```
Using Offers to Reduce Churn
Server-to-server notification

```json
{ notification_type: "DID_CHANGE_RENEWAL_STATUS", ... }
```
# Using Offers to Reduce Churn

## Retention

A Display Subscription Offer was presented between Mar. 26 and Apr. 26.

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>retentionOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000003</td>
<td>4</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>56</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Using Offers to Reduce Churn

Retention

Display Subscription Offer

<table>
<thead>
<tr>
<th></th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>retentionOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>userld</td>
<td>90000003</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Explore with us again.
We’ve noticed you won’t be staying with us next month,
Subscribe now to get another month free.

Get next month for free.
$2.49/mo for 6 months, then $4.99/mo.

Subscribe now
Ultimate Guitar PRO

Get a free month and 50% OFF 🎁

Today Only

OFFER DETAILS

Current plan
$19.99/year

New plan (50% OFF)
$9.99/year

You will not be charged for 1 month. Subscription automatically renews for $9.99/year unless it is canceled at least 24 hours before the end of the 1-month period. Your account will be charged for renewal within 24 hours prior to the end of the current period. You can manage and cancel your subscriptions by going to...

Free for 1 month, then $9.99/year

CONTINUE
Using Offers to Reduce Churn

Upgrade

Mar. 26 - Apr. 26

in_app: [{
  "purchase_date": "2019-03-26...",
  "expires_date": "2019-04-26...",
}]

Using Offers to Reduce Churn

Upgrade

Downgrade Subscription

pending_renewal_info: [{
  "original_transaction_id": "113357799",
  "product_id": "plus_subscription_999_6m",
  "auto_renew_status": "1",
  "auto_renew_product_id": "com.your.product.id",
  ...
}]

Mar. 26 Apr. 1 Apr. 26
Using Offers to Reduce Churn

Upgrade

Downgrade Subscription

Mar. 26  Apr. 1  Apr. 26

pending_renewal_info: [{
  "original_transaction_id": "113357799",
  "product_id": "plus_subscription_999_6m",
  "auto_renew_status": "1",
  "auto_renew_product_id": "com.your.product.id", ...
}]

Mar. 26  Apr. 1  Apr. 26
Auto Renew Preference

pending_renewal_info: [{
    "original_transaction_id": "1133557799",
    "product_id": "plus_subscription_999_6m",
    "auto_renew_status": "1",
    "auto_renew_product_id": "com.your.product.id", ...
}]

Your Server
/processTransaction?userId=90000001

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>autoRenewProductID</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>com.your.product.id</td>
</tr>
</tbody>
</table>
Auto Renew Preference

```json
pending_renewal_info: [{
  "original_transaction_id": "1133557799",
  "product_id": "plus_subscription_999_6m",
  "auto_renew_status": "1",
  "auto_renew_product_id": "com.your.product.id",
}]
```

/processTransaction?userId=90000001

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>autoRenewProductID</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>com.your.product.id</td>
</tr>
</tbody>
</table>
Using Offers to Reduce Churn

Server-to-server notification

```
{ notification_type: "DID_CHANGE_RENEWAL_PREFERENCE", ... }
```
Using Offers to Reduce Churn

Server-to-server notification

We've noticed you selected another product.

{ notification_type: "DID_CHANGE_RENEWAL_PREFERENCE", ... }
Example Subscription

Display Subscription Offer

Mar. 26  Apr. 1  Apr. 26

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000003</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
### Example Subscription

Display Subscription Offer

<table>
<thead>
<tr>
<th>Mar. 26</th>
<th>Apr. 1</th>
<th>Apr. 26</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000003</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Explore more.
Upgrade to annual and save 50% off your total year.

Get 12 months at 50% off.
$2.49/mo for 6 months, then $4.99/mo.

Upgrade now
Your plan: Free
Upgrade to get the best of Dashlane

Premium
Special offer
Get 50% off Premium when you renew today

- Store unlimited passwords on unlimited devices
- Personalized security alerts with Dark Web Monitoring
- VPN for WiFi protection

Dashlane's Premium sync and backup service is available as a 1 year renewable subscription at €39.99:
• Payment will be charged to your iTunes Account at confirmation of purchase
• Auto-renewable subscriptions automatically renew unless auto-renew is turned off at least 24-hours before the end of the current period
• Your account will be charged for renewal within 24-hours before the end of the current period

Get Premium
Using Offers to Reduce Churn

Customer service
Using Offers to Reduce Churn

Customer service

Mar. 26

Apr. 26

in_app: [{
    "purchase_date": "2019-03-26...",
    "expires_date": "2019-04-26...",
}]
Using Offers to Reduce Churn

Customer service

Outage in Service (App)

Mar. 26  Apr. 1  Apr. 26

in_app: [{
   "purchase_date": "2019-03-26...",
   "expires_date": "2019-04-26...",
}]

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>latestExpiresDate</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>contentConsumed</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
<td>0</td>
<td>0</td>
<td>22.5</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
<td>1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>userId</td>
<td>originalTransactionId</td>
<td>latestExpiresDate</td>
<td>autoRenewStatus</td>
<td>billingRetryStatus</td>
<td>contentConsumed</td>
</tr>
<tr>
<td>--------</td>
<td>------------------------</td>
<td>--------------------</td>
<td>-----------------</td>
<td>-------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>90000001</td>
<td>1133557799</td>
<td>2019-07-08...</td>
<td>0</td>
<td>0</td>
<td>22.5</td>
</tr>
<tr>
<td>90000002</td>
<td>4820018625</td>
<td>2019-06-02...</td>
<td>1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>90000003</td>
<td>6090522426</td>
<td>2019-06-01...</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>90000004</td>
<td>3092890314</td>
<td>2019-06-20...</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>90000006</td>
<td>4286343643</td>
<td>2019-06-03...</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
## Example Subscription

### Display Subscription Offer

<table>
<thead>
<tr>
<th>Date</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mar. 26</td>
<td></td>
</tr>
<tr>
<td>Apr. 1</td>
<td></td>
</tr>
<tr>
<td>Apr. 26</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>latestExpiresDate</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>hoursViewed</th>
<th>offerEligible</th>
</tr>
</thead>
<tbody>
<tr>
<td>900000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>
Example Subscription

<table>
<thead>
<tr>
<th>userId</th>
<th>originalTransactionId</th>
<th>latestExpiresDate</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>hoursViewed</th>
<th>offerEligible</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000005</td>
<td>8426576390</td>
<td>2019-06-21...</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>1</td>
</tr>
</tbody>
</table>
Yearly
Unlimited access to Forest Explorer. Your subscription will renew on Dec 1, 2019.

We’re sorry you experienced downtime with your service this month.

Redeem your free month

ACCOUNT INFO
Contact Info
Notifications
Saved Locations

APP INFO
Terms of Service
Keep exploring with us.
We’re sorry you experienced downtime with your service.
You can redeem your free month.

Get next month for free.
$2.49/mo for 6 months, then $4.99/mo.

Redeem your free month
Using Offers to Reduce Churn

Loyalty

Mar. 26

in_app: [{
  "purchase_date": "2019-03-26...",
  "expires_date": "2019-04-26...",
}]

Apr. 26
Using Offers to Reduce Churn

Loyalty

Mar. 26 Apr. 26

11th Consecutive Renewal

pending_renewal_info: [{
  "original_transaction_id": "1133557799",
  "product_id": "plus_subscription_999_6m",
  "auto_renew_status": "1",
}]

## Using Offers to Reduce Churn

Storing consecutive renewals

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>90000003</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Using Offers to Reduce Churn
Storing consecutive renewals

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>90000003</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
## Using Offers to Reduce Churn

Storing consecutive renewals

<table>
<thead>
<tr>
<th>userId</th>
<th>renewalPeriods</th>
<th>autoRenewStatus</th>
<th>billingRetryStatus</th>
<th>billingIssueCount</th>
<th>contentConsumed</th>
<th>winbackOffer</th>
<th>loyaltyOffer</th>
<th>upgradeOffer</th>
</tr>
</thead>
<tbody>
<tr>
<td>90000001</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>90000002</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>125</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>90000003</td>
<td>11</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>352</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>
Using Offers to Reduce Churn

Loyalty

Mar. 26 - Apr. 26

11th Consecutive Renewal

userId | renewalPeriods | autoRenewStatus | billingRetryStatus | billingIssueCount | contentConsumed | winbackOffer | loyaltyOffer | upgradeOffer
--- | --- | --- | --- | --- | --- | --- | --- | ---
90000003 | 11 | 1 | 0 | 2 | 352 | 0 | 1 | 0
Using Offers to Reduce Churn

Loyalty

11th Consecutive Renewal

Mar. 26 - Apr. 26

userId: 90000003
renewalPeriods: 11
autoRenewStatus: 1
billingRetryStatus: 0
billingIssueCount: 2
contentConsumed: 352
winbackOffer: 0
loyaltyOffer: 1
upgradeOffer: 0
Keep exploring with us.
Thank you for being a loyal subscriber.

Get 1 month for free.
$2.49/mo for 6 months, then $4.99/mo.

Redeem your free month
Get 2 more months of Premium free!
As an early Premium subscriber we want to thank you by offering 2 more months of Premium on us. Enjoy!

Activate now
Followed by $7.99/month after

No Thanks

Payment will be charged to your Apple ID account at the confirmation of purchase. The subscription automatically renews unless it is canceled at least 24 hours before the end of the current period. Any unused portion of a free trial period, if offered, is not refundable.
Using Offers to Reduce Churn

Save Offers

Mar. 26 - Apr. 26

in_app: [{
  "purchase_date": "2019-03-26...",
  "expires_date": "2019-04-26...",
}]

...
Using Offers to Reduce Churn

Save Offers

User taps ‘Manage Subscription’

Mar. 26  Apr. 1  Apr. 26
Explore with us again.
Don’t leave, get your next month at half off.

Get next month at 50% off.
$2.49/mo for 6 months, then $4.99/mo.

Resubscribe now
Using Offers to Reduce Churn

Reporting

New reporting dashboards

• Subscription State
• Subscription Events
Using Offers to Reduce Churn

Reporting

New reporting dashboards

• Subscription State
• Subscription Events

Analyze subscription activity
4.82K Active Subscriptions

State

<table>
<thead>
<tr>
<th>State</th>
<th>Ending on May 30, 2019</th>
<th>Previous 30 Days</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>Promotional Offer</td>
<td>4.82K</td>
<td>0</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Implement Subscription Offers
Summary

Implement Subscription Offers

• In your app and on your server
Implement Subscription Offers

• In your app and on your server

Implement customer eligibility
Summary

Implement Subscription Offers
• In your app and on your server

Implement customer eligibility
• Receipt
Summary

Implement Subscription Offers
• In your app and on your server

Implement customer eligibility
• Receipt
• Sever-to-server notifications
Summary

Implement Subscription Offers
• In your app and on your server

Implement customer eligibility
• Receipt
• Sever-to-server notifications
• Developer data
Summary

Implement Subscription Offers
• In your app and on your server

Implement customer eligibility
• Receipt
• Sever-to-server notifications
• Developer data

Present subscription offers to customers contextually
Summary

Implement Subscription Offers
• In your app and on your server

Implement customer eligibility
• Receipt
• Sever-to-server notifications
• Developer data

Present subscription offers to customers contextually

Monitor performance of offers
More Information

developer.apple.com/wwdc19/305