Making the Most of Search APIs

New features and updates expand search capabilities

Session 223

Vipul Ved Prakash  Apple, Inc.
John Hörnkvist  Apple, Inc.
Spotlight Updates in iOS 10

Notification Center
Note from Max

*NEW*

Search from New Lockscreen

NEW
Note from Max

*NEW*

Search from New Lockscreen

NEW
I'm at the store. What do we need for lettuce wraps?
This easy appetizer is packed with flavor and crunch and comes together in less than half an hour. That means less time in the kitchen and more time enjoying long summer evenings! Spoon the mixture into halved mini sweet peppers to serve it as a festive party dish.

**NUTRITIONAL INFORMATION**

Per Serving: 230 calories (70 from fat), 7g total fat, 65mg cholesterol, 530mg sodium, 33g carbohydrate (2g dietary fiber, 3g sugar), 8g protein

**INGREDIENTS**

1 box 365 Everyday Value® frozen Lightly Breaded Calamari

2 teaspoons expeller-pressed canola oil

1/2 pound (8 ounces) green beans, thinly sliced

1/2 package frozen Whole Foods Market Vegetable Fried Rice
Spotlight Updates in iOS 10

Notification Center

Search on Lock Screen

3D Touch
SPORTS
Stephen Curry
#30, Point Guard, Golden State Warriors
6' 3'', 190 lb
NBA

WIKIPEDIA
Stephen Curry
Wardell Stephen "Steph" Curry II (born March 14, 1988) is an American professional basketball player for the...

WEB VIDEOS
Stephen Curry Is The Worst MVP To Ever Play...
by CashNastyGaming
6/8/2016 • 8.5K views
YouTube

I GOT INTO A BLOODY FIGHT WITH STEPHEN...
by CashNastyGaming
6/7/2016 • 31K views
YouTube
Spotlight Updates in iOS 10

Notification Center
Search on Lock Screen
3D Touch
Query Suggestions
Search APIs Released in iOS 9

Three technologies for implementing search
Search APIs Released in iOS 9

Three technologies for implementing search

CoreSpotlight

Any App Content
Search APIs Released in iOS 9

Three technologies for implementing search

- **CoreSpotlight**: Any App Content
- **NSUserActivity**: Viewed App Content
Search APIs Released in iOS 9

Three technologies for implementing search

- CoreSpotlight: Any App Content
- NSUserActivity: Viewed App Content
- Universal Links: App Content on Web
50,000+
Apps using Search APIs
Kayak Travel Services
Flight Reservation #20053423
Thank you for booking your trip to Maui, HI with Kayak.com. Login, See Services, Touring...

OpenTable
Reservation Confirmation
Thank you for booking your trip to Maui, HI with Kayak.com. Login, See Services, Touring...

Maui
The island of Maui is the second largest of the Hawaiian Islands at 7272 square miles and is the 17th largest of the Hawaiian...
Hey, Vipul!
Hope you’re looking forward to your visit!

Aston Kaanapali Villas
45 Kaai Ala Drive, Maui, HI
Sat June 18 - Fri June 24

Forecast: Some ☁
80°F | 69°F

Why We Like It
Close to a wealth of outdoor adventure – hiking on hills, whale watching on a catamaran, snorkeling among the maritime fauna, all nearby.

Check-in
Sat June 18, 2016
3:00 PM

Check-out
Fri June 24, 2016
12:00 PM
TOP 20 ACTIVITIES ON MAUI

About City of Hana, HI

20 Best Activities Maui has to offer
20 Best Activities #Maui has to offer!

Kate Johnson

Saved to Oh, the places you'll go...

Add a comment

10 of Hawaii's Best Hidden Beaches

msn.com

Anita Clay

All Who Wander are...

The 3 Most Awe-Inspiring Hikes in Maui

The Huffington Post

Heather Grause

Aloha

TOP 10 SNORKELING SPOTS IN MAUI
Content Your Users Want to Get Back To

Order confirmations
Upcoming events
Tickets
Reservations
Deliveries
Hi Vipul!

Hope all is well with you. I am coming home from London and was wondering if you might be able to pick me up from the airport? My flight lands at 1:20pm in SFO. Flight is United Airlines 900.

Had a great time, but looking forward to getting home. Hope you enjoy the photo below. BTW my phone number is now (415) 555-9552.

Thanks!
Allen
Hi Vipul

Hope you are doing great!
Enhance Native Spotlight Content
Enhance Native Spotlight Content

<table>
<thead>
<tr>
<th>Instant answers</th>
<th>Sports</th>
<th>Web search</th>
<th>Music</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contacts</td>
<td>Locations</td>
<td>Apps</td>
<td>TV</td>
</tr>
<tr>
<td>Stocks</td>
<td>Weather</td>
<td>Calculations</td>
<td>Movies</td>
</tr>
</tbody>
</table>
WEB VIDEOS

How to Perform CPR
by CPRCertified.com
Uploaded 2/10/2015 • 28M views
YouTube

WIKIHOW

How to Do CPR on an Adult
Knowing how to perform both methods of CPR (cardiopulmonary resuscitation) on an adult could save a life. However, the recommended...

How to Do CPR on a Baby
Though CPR (cardiopulmonary resuscitation) should be administered by individuals trained at a certified first aid course, normal bystanders c...

How to Perform Self CPR
CPR (cardiopulmonary resuscitation) is a lifesaving technique that’s useful in many emergency situations, such as heart attacks...

Search Web
Search App Store
Search Maps
How to Do CPR on an Adult

In 2010, the American Heart Association made a radical change to the recommended CPR process for victims of cardiac arrest [1] after studies showed that compression-only CPR (no mouth-to-mouth breathing) is as effective as the traditional approach. Here's how to perform both methods of CPR on an adult.

Method One of Four
Taking Vitals

1. Check the scene for immediate
What’s New

New in iOS 10
What’s New

New in iOS 10

Search in App
CoreSpotlight Search API
Differential Privacy
Visual Preview in Web Markup Tool
Katie Ables
Trip to San Francisco
Vipul, Visiting you in San Francisco was so much fun. We definitely need to head back...

Ethan Izzarelli
Surfing Trip
Hey Vipul, Our surfing trip to Hawaii was incredible. We stayed on the north shore...

Derek Parker
Australia Trip
Hi Vipul, Our family vacation to Australia was incredible. We visited Sydney, Brisbane...

Trips around the world
Sanaa Lathan I love travel. These are dream images. 569 Pins

How to Plan a Backpacking Trip
With a little effort, backpacking trips can be serious fun. A well-planned trip will all...
Users can continue their Search in the App.
John Daily  
Yosemite Trip  
Hey Vipul, We're getting excited for our trip to Yosemite this summer. The kids are going to h...

Katie Abeles  
Trip to San Francisco  
Vipul, Visiting you in San Francisco was so much fun. We definitely need to head back soon. Her...

Derek Parker  
Australia Trip  
Hi Vipul, Our family vacation to Australia was incredible. We visited Sydney, Brisbane, and M...

Ethan Izzarelli  
Surfing Trip  
Hey Vipul, Our surfing trip to Hawaii was incredible. We stayed on the north shore of Oa...

Emily Bergendahl  
Walking the Golden Gate  
Vipul, Our trip to San Francisco was so much fun. We biked over the Golden Gate Bridge on...

Natalia Maric  

What’s New

New in iOS 10

Search in App
CoreSpotlight Search API
Differential Privacy
Visual Preview in Web Markup Tool
What’s New

New in iOS 10

Search in App
CoreSpotlight Search API
Differential Privacy
Visual Preview in Web Markup Tool
What’s New

New in iOS 10

Search in App
CoreSpotlight Search API
Differential Privacy
Visual Preview in Web Markup Tool
What’s New

New in iOS 10

Search in App
CoreSpotlight Search API
Differential Privacy
Visual Preview in Web Markup Tool
Adopting Search APIs
Making the most of your content

John Hörnkvist Apple, Inc.
Adopting Search APIs
Steps and best practices

Indexing content
Rich presentation
Launching into your app
CoreSpotlight Search API
Optimize for ranking
CoreSpotlight

Any App Content
CoreSpotlight
Any App Content

NSUserActivity
Viewed App Content
CoreSpotlight
Any App Content

NSUserActivity
Viewed App Content

Universal Links
App Content on Web
Black Quinoa & Kale Salad
Salads, Gluten free, Whole grain, Serves 4, Cooking time 25 min 5 ingredients

Grilled Asparagus & Creamy Lemon Quinoa
Main courses, Gluten free, Serves 4, Cooking time 45 min, 5 ingredients

Black Quinoa & Kale Salad
Salads, Gluten free, Whole grain, Serves 4, Cooking time 25 min 5 ingredients

Quinoa
Quinoa is a species of the goosefoot genus (Chenopodium quinoa), a grain crop grown primarily for its edible seed.
Adopting Search APIs
Steps and best practices

Indexing content
Rich presentation
Launching into your app
CoreSpotlight Search API
Optimize for ranking
Indexing with CoreSpotlight

Index all that your app has to offer

API for on-device indexing on iOS

You decide what you want to add to the index

Documents, game levels, images, jump-off points, and more
CoreSpotlight Refresher
Basic operations

Adding items to the index

```swift
let attrs : CSSearchableItemAttributeSet = 
    CSSearchableItemAttributeSet(itemContentType: kUTTypeItem as String)
attrs.displayName = "Hello world!"
let item = CSSearchableItem(uniqueIdentifier:"hello", domainIdentifier:
    "Greetings", attributeSet: attrs)
CSSearchableIndex.defaultIndex().indexSearchableItems([item],
    completionHandler: handler)
```
Adding items to the index

```swift
let attrs : CSSearchableItemAttributeSet = 
    CSSearchableItemAttributeSet(itemContentType: kUTTypeItem as String)

attrs.displayName = "Hello world!"

let item = CSSearchableItem(uniqueIdentifier:"hello", domainIdentifier: 
    "Greetings", attributeSet: attrs)

CSSearchableIndex.defaultIndex().indexSearchableItems([item], 
    completionHandler: handler)
```
Core Spotlight Refresher
Basic operations

Adding items to the index

```swift
let attrs: CSSearchableItemAttributeSet = 
    CSSearchableItemAttributeSet(itemContentType: kUTTypeItem as String)
attrs.displayName = "Hello world!"

let item = CSSearchableItem(uniqueIdentifier:"hello", domainIdentifier: 
    "Greetings", attributeSet: attrs)

CSSearchableIndex.defaultIndex().indexSearchableItems([item], 
    completionHandler: handler)
```
CoreSpotlight Refresher

Basic operations

Adding items to the index

```swift
let attrs : CSSearchableItemAttributeSet =
    CSSearchableItemAttributeSet(itemContentType: kUTTypeItem as String)
attrs.displayName = "Hello world!"

let item = CSSearchableItem(uniqueIdentifier:"hello", domainIdentifier:
    "Greetings", attributeSet: attrs)

CSSearchableIndex.defaultIndex().indexSearchableItems([item],
    completionHandler: handler)
```
CoreSpotlight Refresher
Basic operations

Adding items to the index

```swift
let attrs : CSSearchableItemAttributeSet = 
    CSSearchableItemAttributeSet(itemContentType: kUTTypeItem as String)
attrs.displayName = "Hello world!"
let item = CSSearchableItem(uniqueIdentifier:"hello", domainIdentifier:
    "Greetings", attributeSet: attrs)

CSSearchableIndex.defaultIndex().indexSearchableItems([item],
    completionHandler: handler)
```
Deleting items from the index

```swift
CSSearchableIndex.defaultIndex().deleteSearchableItems(withIdentifiers:
    "hello",
    completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteSearchableItems(withDomainIdentifiers:
    "Greetings",
    completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteAllSearchableItems(completionHandler:handler)
```
Deleting items from the index

```swift
CSSearchableIndex.defaultIndex().deleteSearchableItems(withIdentifiers: ["hello"], completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteSearchableItems(withDomainIdentifiers: ["Greetings"], completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteAllSearchableItems(completionHandler: handler)
```
Deleting items from the index

```swift
CSSearchableIndex.defaultIndex().deleteSearchableItems(withIdentifiers: ["hello"], completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteSearchableItems(withDomainIdentifiers: ["Greetings"], completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteAllSearchableItems(completionHandler: handler)
```
Deleting items from the index

```swift
CSSearchableIndex.defaultIndex().deleteSearchableItems(withIdentifiers: ["hello"], completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteSearchableItems(withDomainIdentifiers: ["Greetings"], completionHandler: handler)

CSSearchableIndex.defaultIndex().deleteAllSearchableItems(completionHandler: handler)
```
Indexing with CoreSpotlight

Best practices

Registering as an index delegate
Use client state API for asynchronous indexing
Performance considerations
Creating a CoreSpotlight extension
Indexing with CoreSpotlight

Register as an index delegate

Spotlight will reach out to initiate indexing

```swift
CSSearchableIndex.defaultSearchableIndex().indexDelegate = self
```
Indexing with CoreSpotlight

Implement CSSearchableIndexDelegate protocol

```swift
func searchableIndex(_: CSSearchableIndex,
    reindexAllSearchableItemsWithAcknowledgementHandler acknowledgementHandler: () -> Void) {
    // Reindex everything, then call handler
    acknowledgementHandler();
}

func searchableIndex(_: CSSearchableIndex,
    reindexSearchableItemsWithIdentifiers identifiers: [String],
    acknowledgementHandler: () -> Void) {
    // Reindex items for the identifiers, then call handler
    acknowledgementHandler()
}
```
Indexing with CoreSpotlight

Implement CSSearchableIndexDelegate protocol

```swift
func searchableIndex(_: CSSearchableIndex,
    reindexAllSearchableItemsWithAcknowledgementHandler acknowledgementHandler: () -> Void) {
    // Reindex everything, then call handler
    acknowledgementHandler();
}

func searchableIndex(_: CSSearchableIndex,
    reindexSearchableItemsWithIdentifiers identifiers: [String],
    acknowledgementHandler: () -> Void) {
    // Reindex items for the identifiers, then call handler
    acknowledgementHandler()
}
```
Indexing with CoreSpotlight

Implement CSSearchableIndexDelegate protocol

```swift
func searchableIndex(_: CSSearchableIndex,
    reindexAllSearchableItemsWithAcknowledgementHandler acknowledgementHandler: () -> Void) {
    // Reindex everything, then call handler
    acknowledgementHandler();
}

func searchableIndex(_: CSSearchableIndex,
    reindexSearchableItemsWithIdentifiers identifiers: [String],
    acknowledgementHandler: () -> Void) {
    // Reindex items for the identifiers, then call handler
    acknowledgementHandler();
}
```
Indexing with CoreSpotlight

Implement CSSearchableIndexDelegate protocol

```swift
func searchableIndex(_: CSSearchableIndex,
    reindexAllSearchableItemsWithAcknowledgementHandler acknowledgementHandler: () -> Void) {
    // Reindex everything, then call handler
    acknowledgementHandler();
}

func searchableIndex(_: CSSearchableIndex,
    reindexSearchableItemsWithIdentifiers identifiers: [String],
    acknowledgementHandler: () -> Void) {
    // Reindex items for the identifiers, then call handler
    acknowledgementHandler()
}
```
Indexing with CoreSpotlight

Using client state for asynchronous updates

Allows you to keep your datastore and Spotlight in sync, without blocking

An opaque token stored in Spotlight’s index

Used with journals or database annotations
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source

```swift
let index = CSSearchableIndex(name: "myname")

index.beginBatch()

index.indexSearchableItems(items, completionHandler: nil);

let stateString = String(offset + items.count)

index.endBatch(withClientState: stateString.data(using:NSUTF8StringEncoding)!, completionHandler: handler)
```
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source

```swift
let index = CSSearchableIndex(name: "mynamen")

index.beginBatch()

index.indexSearchableItems(items, completionHandler: nil);

let stateString = String(offset + items.count)

index.endBatch(withClientState: stateString.data(using:NSUTF8StringEncoding)!, completionHandler: handler)
```
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source

```swift
let index = CSSearchableIndex(name: "myname")

index.beginBatch()

index.indexSearchableItems(items, completionHandler: nil);

let stateString = String(offset + items.count)

index.endBatch(withClientState: stateString.data(using:NSUTF8StringEncoding)!, completionHandler: handler)
```
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source

```swift
let index = CSSearchableIndex(name: "mymame")

index.beginBatch()

index.indexSearchableItems(items, completionHandler: nil);

let stateString = String(offset + items.count)

index.endBatch(withClientState: stateString.data(using:NSUTF8StringEncoding)!,
completionHandler: handler)
```
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source

```swift
let index = CSSearchableIndex(name: "myname")

index.beginBatch()

index.indexSearchableItems(items, completionHandler: nil);

let stateString = String(offset + items.count)

index.endBatch(withClientState: stateString.data(using:NSUTF8StringEncoding)!, completionHandler: handler)
```
Indexing with CoreSpotlight

Setting client state

Index state uses a named index instance
Create multiple instances if you have more than one data source

```swift
let index = CSSearchableIndex(name: "mynamen")

index.beginBatch()

index.indexSearchableItems(items, completionHandler: nil);

let stateString = String(offset + items.count)

index.endBatch(withClientState: stateString.data(using:NSUTF8StringEncoding)!, completionHandler: handler)
```
Indexing with CoreSpotlight

Checking client state

Fetch client state on launch
Resume interrupted indexing
func indexIfNeeded()
{
    let index = CSSearchableIndex(name: "myname")

    index.fetchLastClientState(completionHandler:  { (data, error) in
        if (error != nil) {
            // deal with the error.
        } else if (data != expectedData) {
            // Index state wasn’t what we expected. Bring CoreSpotlight up to date.
            self.doIndex(index, data)
        }
    })
}

// Call on launch to fetch index state and start indexing if needed.
// ** Call on launch to fetch index state and start indexing if needed.
func indexIfNeeded()
{
    // ** Create a named index instance.
    let index = CSSearchableIndex(name: "myname")

    // ** Asynchronously fetch our last client state.
    index.fetchLastClientState(completionHandler: { (data, error) in
        if (error != nil) {
            // deal with the error.
        } else if (data != expectedData) {
            // Index state wasn’t what we expected. Bring CoreSpotlight up to date.
            self.doIndex(index, data)
        }
    })
}
func indexIfNeeded()
{
// ** Create a named index instance.
let index = CSSearchableIndex(name: "myname")

// ** Asynchronously fetch our last client state.
index.fetchLastClientState(completionHandler: { (data, error) in
    if (error != nil) {
        // deal with the error.
    } else if (data != expectedData) {
        // Index state wasn’t what we expected. Bring CoreSpotlight up to date.
        self.doIndex(index, data)
    }
})
}
// ** Call on launch to fetch index state and start indexing if needed.
func indexIfNeeded() {
    // ** Create a named index instance.
    let index = CSSearchableIndex(name: "myname")

    // ** Asynchronously fetch our last client state.
    index.fetchLastClientState(completionHandler: { (data, error) in
        if (error != nil) {
            // deal with the error.
        } else if (data != expectedData) {
            // Index state wasn’t what we expected. Bring CoreSpotlight up to date.
            self.doIndex(index, data)
        }
    })
}
Indexing with CoreSpotlight

Performance considerations

Minimize overhead, optimize storage and database access
Use batching, size batches for memory overhead
Don’t block the main thread
Index on a background thread
CoreSpotlight Extension
Catching up in the background
The extension can index when your app isn’t running
Recover after restore from backup
CoreSpotlight Extension
Catching up in the background

The extension can index when your app isn’t running
Recover after restore from backup

```swift
func searchableIndex(_: CSSearchableIndex,
    reindexAllSearchableItemsWithAcknowledgementHandler acknowledgementHandler: () -> Void) {
    // Reindex everything, then call handler
    acknowledgementHandler();
}

func searchableIndex(_: CSSearchableIndex,
    reindexSearchableItemsWithIdentifiers identifiers: [String],
    acknowledgementHandler: () -> Void) {
    // Reindex items for the identifiers, then call handler
    acknowledgementHandler()
}
```
CoreSpotlight Extension
Catching up in the background

The extension can index when your app isn’t running
Recover after restore from backup
Keeping Content Up to Date

Set expiration dates on items that have a natural life span

- CoreSpotlight will call your app to validate expiring item

Use Background Fetch to poll for new content

Use Silent Remote Notifications for push
Keeping Content Up to Date

Set expiration dates on items that have a natural life span

- CoreSpotlight will call your app to validate expiring item

Use Background Fetch to poll for new content

Use Silent Remote Notifications for push

What’s New with Multitasking
When to Submit NSUserActivities?

Adopting Handoff on iOS and OS X

WWDC 2014
Indexing with NSUserActivity

NSUserActivity

Designate activities as searchable
Add metadata for search and display
Indexing with NSUserActivity

NSUserActivity

Designate activities as searchable
Add metadata for search and display

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");

userActivity.eligibleForSearch = true

userActivity.title = "title"

let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = ...

userActivity.contentAttributeSet = attributes
```
Indexing with NSUserActivity

NSUserActivity

Designate activities as searchable
Add metadata for search and display

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");

userActivity.eligibleForSearch = true

userActivity.title = "title"

let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = ...

userActivity.contentAttributeSet = attributes
```
Indexing with NSUserActivity

NSUserActivity

Designate activities as searchable
Add metadata for search and display

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");

userActivity.eligibleForSearch = true

userActivity.title = "title"

let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = …

userActivity.contentAttributeSet = attributes
```
GREEN KITCHEN

**Foamy Leek & Sunchoke Soup**
- Soups, Gluten free, Vegan
- Serves 4
- Cooking time 1 hour 8 ingredients

**Cauliflower Leek Soup**
- Soups, Gluten free, Vegan
- Serves 4
- Cooking time 40 min 12 ingredients

WEB VIDEOS

**English Onion & Leek Soup - Jamie at Home**
- by Jamie Oliver
- 2/12/2013 • 237K views

**Potato Leek Soup from Garden Fresh Vegetables**
- by Brandon & Meredith
- 6/9/2013 • 29K views

**Leek and Potato Soup Recipe | Marco Pierre W...**
- by KnorrRecipes
- 1/6/2011 • 150K views
NSUserActivity and CSSearchableItem

Tying them together

NSUserActivity reflects what the user did
CSSearchableItem reflects what your app has
relatedUniqueIdentifier ties them together
NSUserActivity and CSSearchableItem

Tying them together

NSUserActivity reflects what the user did
CSSearchableItem reflects what your app has
relatedUniqueldentifier ties them together

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");
let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = "Private content!"
attributes.relatedUniqueIdentifier = "myIdentifier"

userActivity.contentAttributeSet = attributes
```
NSUserActivity and CSSearchableItem

Tying them together

NSUserActivity reflects what the user did
CSSearchableItem reflects what your app has
relatedUniqueIdentifier ties them together

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");
let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = "Private content!"
attributes.relatedUniqueIdentifier = "myIdentifier"
userActivity.contentAttributeSet = attributes
```
NSUserActivity and CSSearchableItem

Tying them weakly

New in iOS 10: weakRelatedUniqueIdentifier

• relatedUniqueIdentifier binds strongly
• weakRelatedUniqueIdentifier binds weakly

A weakly bound NSUserActivity can exist without a matching CSSearchableItem

Not suitable for user controlled content
NSUserActivity and CSSearchableItem
Tying them weakly

New in iOS 10: weakRelatedUniqueIdentifier
- relatedUniqueIdentifier binds strongly
- weakRelatedUniqueIdentifier binds weakly

A weakly bound NSUserActivity can exist without a matching CSSearchableItem
Not suitable for user controlled content

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");
let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = "Poutine"

attributes.weakRelatedUniqueIdentifier = "myIdentifier"

userActivity.contentAttributeSet = attributes
```
NSUserActivity and CSSearchableItem

Tying them weakly

New in iOS 10: weakRelatedUniqueIdentifier

- relatedUniqueIdentifier binds strongly
- weakRelatedUniqueIdentifier binds weakly

A weakly bound NSUserActivity can exist without a matching CSSearchableItem

Not suitable for user controlled content

let userActivity = NSUserActivity(activityType: "myActivityType");
let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.displayName = "Poutine"

attributes.weakRelatedUniqueIdentifier = "myIdentifier"

userActivity.contentAttributeSet = attributes
NSUserActivity Deletion

domainIdentifier for NSUserActivity

New in iOS 10: Use domainIdentifier with NSUserActivity
Part of CSSearchableItemAttributeSet
Delete user activities in bulk by domainIdentifier
New in iOS 10: Use domainIdentifier with NSUserActivity
Part of CSSearchableItemAttributeSet
Delete user activities in bulk by domainIdentifier

```swift
let userActivity = NSUserActivity(activityType: "myActivityType");
let attributes = CSSearchableItemAttributeSet(itemContentType: "public.item")
attributes.domainIdentifier = "account1"
userActivity.contentAttributeSet = attributes
```
NSUserActivity Deletion

domainIdentifier for NSUserActivity

New in iOS 10: Use domainIdentifier with NSUserActivity

Part of CSSearchableItemAttributeSet

Delete user activities in bulk by domainIdentifier

```swift
let ci = CSSearchableIndex.defaultSearchableIndex();

ci.deleteSearchableItems(withDomainIdentifiers: ["account1"]) { error in
    // ..
}
```
CoreSpotlight
Any App Content

NSUserActivity
Viewed App Content

Universal Links
App Content on Web
Indexing Universal Links
Content available on the web and in app

Index public content, including large indexes
Reach new users—results shown even if users do not have your app
Results shown in Spotlight and Safari
## Indexing Universal Links

### Steps

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Allow Indexing</td>
<td>Allow Applebot, include domain in iTunes Connect</td>
</tr>
<tr>
<td>2. Deep Linking</td>
<td>Support Universal Links and Smart App Banners</td>
</tr>
<tr>
<td>3. Markup</td>
<td>Markup website with schema.org and Open Graph</td>
</tr>
<tr>
<td>4. Restoring</td>
<td>Update app to properly open deep links</td>
</tr>
<tr>
<td>5. Test</td>
<td>Test URLs with App Search API Validation Tool</td>
</tr>
</tbody>
</table>
# Indexing Universal Links

## Steps

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td><strong>Allow Indexing</strong>&lt;br&gt;Allow Applebot, include domain in iTunes Connect</td>
</tr>
<tr>
<td>2.</td>
<td><strong>Deep Linking</strong>&lt;br&gt;Support Universal Links and Smart App Banners</td>
</tr>
<tr>
<td>3.</td>
<td><strong>Markup</strong>&lt;br&gt;Markup website with schema.org and Open Graph</td>
</tr>
<tr>
<td>4.</td>
<td><strong>Restoring</strong>&lt;br&gt;Update app to properly open deep links</td>
</tr>
<tr>
<td>5.</td>
<td><strong>Test</strong>&lt;br&gt;Test URLs with App Search API Validation Tool</td>
</tr>
</tbody>
</table>

---

Support Universal Links—iOS Developer Library

Introducing Search APIs
Indexing Universal Links

Supported Schemas

AggregateRating
InteractionCount
Offers
PriceRange
Organization
Recipe
SearchAction
ImageObject
Visual Preview in API Validation Tool

https://search.developer.apple.com
Visual Preview in API Validation Tool

https://search.developer.apple.com
Visual Preview in API Validation Tool

App Search API Validation Tool

Test your webpage for iOS 9 Search APIs. Enter a URL and Applebot will crawl your webpage and show you how you can optimize for best results. Learn more about Search for Developers.

https://www.tripadvisor.com/Restaurant_Review-g60713-d4715271-Reviews-Shorty_Goldstein_s-San_Francisco_

Element | Status | Extracted Data |
---|---|---|
Title | | Shorty Goldstein’s, San Francisco - Financial District -…
Description | | Shorty Goldstein’s, San Francisco: See 12 unbiased reviews of Shorty Goldstein’s, rated 4.5 of 5 on TripAdvisor and ranked #1,227 of 5,499 restaurants in San Francisco.
Image | | https://static.tacdn.com/img2/branding/TA_550x370.png
Visual Preview in API Validation Tool

App Search API Validation Tool

Test your webpage for iOS 9 Search APIs. Enter a URL and Applebot will crawl your webpage and show how you can optimize for best results. Learn more about Search for Developers.


[Image of the tool's interface with elements labeled: Title, Description, Markup]
Tying Them All Together

CoreSpotlight

NSUserActivity

Universal Links
Tying Them All Together

NSUserActivity
.contentAttributeSet
.relatedUniqueIdentifier = CSSearchableItem.
uniqueIdentifier

CoreSpotlight

Universal Links
Tying Them All Together

NSUserActivity
.contentAttributeSet
.relatedUniqueIdentifier = CSSearchableItem.
uniqueIdentifier

CoreSpotlight

CSSearchableItem.contentURL = Web page URL

Universal Links
Tying Them All Together

NSUserActivity
- .contentAttributeSet
- .relatedUniqueIdentifier = CSSearchableItem.
  uniqueIdentifier

CoreSpotlight

CSSearchableItem.contentURL = Web page URL

Universal Links

NSUserActivity.webpageURL = Web Page URL
Adopting Search APIs
Steps and best practices

Indexing content
Rich presentation
Launching into your app
CoreSpotlight Search API
Optimize for ranking
Creating a Great Visual Experience

Provide descriptive title, description, and compelling thumbnail

Set the right contentType for your content

Use attributes to fill out the UI

<table>
<thead>
<tr>
<th>Element</th>
<th>CoreSpotlight</th>
<th>Web Markup</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thumbnail</td>
<td>thumbnailURL</td>
<td>og:image</td>
</tr>
<tr>
<td>Title</td>
<td>title</td>
<td>og:title</td>
</tr>
<tr>
<td>Description</td>
<td>contentDescription</td>
<td>og:description</td>
</tr>
<tr>
<td>Metadata</td>
<td>rating, ratingDescription, completionDate, dueDate, startDate, endDate, allDay, fileSize, pageCount</td>
<td>aggregateRating, InteractionCount, Offers, PriceRange, Organization, Recipe, SearchAction, ImageObject</td>
</tr>
</tbody>
</table>
Creating a Great User Experience

Make it easy to get your content
• Set attributes the user can understand
• Keyword stuffing confuses the user and leads to poor ranking
Enable quick actions, like directions and calling
Launch straight to content, without interstitials or multi-step builds
 postalAddress

 telephone
Adopting Search APIs
Steps and best practices

Indexing content
Rich presentation
Launching into your app
CoreSpotlight Search API
Optimize for ranking
Restoring with NSUserActivity

Used for CoreSpotlight and NSUserActivity
func application(application: UIApplication, continueUserActivity uA: NSUserActivity, restorationHandler: ([AnyObject]?) -> Void) -> Bool {
    if uA.activityType == CSSearchableItemActionType {
        if let i = uA.userInfo?[CSSearchableItemActivityIdentifier] as? String {
            // Show the found item!
        }
        return true
    }
    return false
}
Restoring Universal Links

Universal Link support in your app delegate
func application(application: UIApplication, 
    continueUserActivity uA: NSUserActivity, 
    restorationHandler: ([AnyObject]?) -> Void) -> Bool {

    if uA.activityType == NSUserActivityTypeBrowsingWeb {
        if let url = uA.webPageURL {
            if let components = NSURLComponents(url: uA.webPageURL, 
                resolvingAgainstBaseURL: true) {
                let path = components.path, let query = components.query {
                    // Show the found item!
                }
            }
        }
        return true
    }
    return false
}
Continue Search in App

Search in App brings users back to your app
Simple adoption
Leverage your customized search interface
Widely adopted in system apps
NEW
Users can continue their Search in the App.
John Daily
Yosemite Trip
Hey Vipul, We're getting excited for our trip to Yosemite this summer. The kids are going to h...

Katie Abeles
Trip to San Francisco
Vipul, Visiting you in San Francisco was so much fun. We definitely need to head back soon. Her...

Derek Parker
Australia Trip
Hi Vipul, Our family vacation to Australia was incredible. We visited Sydney, Brisbane, and M...

Ethan Izzarelli
Surfing Trip
Hey Vipul, Our surfing trip to Hawaii was incredible. We stayed on the north shore of Oa...

Emily Bergendahl
Walking the Golden Gate
Vipul, Our trip to San Francisco was so much fun. We biked over the Golden Gate Bridge on...

Natalia Maric
Enabling Search Continuation

Declare that your app supports Search Continuation
Add a `CoreSpotlightContinuation` key in your Info.plist
Implement user activity support
Enabling Search Continuation

Declare that your app supports Search Continuation

Add a `CoreSpotlightContinuation` key in your Info.plist

Implement user activity support

```swift
func application(application: UIApplication, continueUserActivity userActivity: NSUserActivity, restorationHandler: ([AnyObject]?) -> Void) -> Bool {
    if userActivity.activityType == CSQueryContinuationActionType {
        if let searchQuery = userActivity.userInfo?[CSSearchQueryString] as? String {
            // Invoke your search UI!
        }
        return true
    }
    return false
}
```
Adopting Search APIs
Steps and best practices

Indexing content
Rich presentation
Launching into your app
CoreSpotlight Search API
Optimize for ranking
Introducing CoreSpotlight Search API

Search your Spotlight index

Same search that powers Spotlight, Mail, Notes, and more

One index—great for power and performance

Combine with your own network search for responsiveness and completeness
Search for “Trip”

People
- Montana Triplet
- David Triptree
- Sender contains: Trip

Subjects
- Subject contains: Trip
- Trip to San Francisco
- Maui trip - flight booking - landing page
- Biz trip like a boss
- Yosemite Trip
- Australia Trip
- Surfing Trip
- More Savings for Your Oakland Trip!
CoreSpotlight Search API

API for your application to query its own data in the Spotlight index
Fast and scalable
Simple, easy to use API
Powerful query syntax

playCount > 1 && (composer == "beethoven" || composer="mozart") && InRange(tempo, 40, 80)

* = "bob*" && textContent="project updates"
CoreSpotlight Search API

API for your application to query it’s own data in the Spotlight index
Fast and scalable
Simple, easy to use API
Powerful query syntax

playCount > 1 && (composer == “beethoven” || composer=“mozart”) && InRange(tempo, 40, 80)

* = “bob*” && textContent=“project updates”

NEW
Query Syntax

* == “johnny appleseed”

<table>
<thead>
<tr>
<th>Feature</th>
<th>Token</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Equality</td>
<td>==</td>
<td>keywords==“search”</td>
</tr>
<tr>
<td>Not Equal</td>
<td>!=</td>
<td>keywords!=“search”</td>
</tr>
<tr>
<td>Greater than</td>
<td>&gt;, &gt;=</td>
<td>pageCount &gt; 10</td>
</tr>
<tr>
<td>Less than</td>
<td>&lt;, &lt;=</td>
<td>pageCount &lt; 10</td>
</tr>
<tr>
<td>Range search</td>
<td>InRange</td>
<td>InRange(pageCount, 5, 10)</td>
</tr>
<tr>
<td>AND</td>
<td>&amp;&amp;</td>
<td>fileSize &gt; 100 &amp;&amp; pageCount &gt; 10</td>
</tr>
<tr>
<td>OR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>NOT</td>
<td>!</td>
<td>!(fileSize &gt; 100</td>
</tr>
<tr>
<td>Field wildcard</td>
<td>*</td>
<td>* == “search”</td>
</tr>
<tr>
<td>Field or content wildcard</td>
<td>**</td>
<td>** == “search”</td>
</tr>
</tbody>
</table>
## String Matching

* `* == “johnny appleseed”cwdt`

<table>
<thead>
<tr>
<th>Feature</th>
<th>Flag</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case insensitive</td>
<td>‘c’</td>
</tr>
<tr>
<td>Diacritics insensitive</td>
<td>‘d’</td>
</tr>
<tr>
<td>Word matching</td>
<td>‘w’</td>
</tr>
<tr>
<td>Tokenized</td>
<td>‘t’</td>
</tr>
</tbody>
</table>
func search(userQuery : String) {
    query.cancel();

    let escapedString = userQuery.replacingOccurrences(of: "\", with: "\\\")
        .replacingOccurrences(of: "\\", with: "\\\\")
    let queryString = "**=" + escapedString + "cwdt"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])

    query.foundItemsHandler = { (items : [CSSearchableItem]) -> Void in
        /* process received items */
    }

    query.completionHandler = { (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }

    query.start()
}
func search(userQuery : String) {
    query.cancel();
    let escapedString = userQuery.replacingOccurrences(of: "\", with: "\\\")
        .replacingOccurrences(of: "\\", with: "\\\\")
    let queryString = "**=" + escapedString + ""cw"t"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])

    query.foundItemsHandler = { (items : [CSSearchableItem]) -> Void in
        /* process received items */
    }

    query.completionHandler = { (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }

    query.start()
}
func search(userQuery : String) {
    query.cancel();
    let escapedString = userQuery.replacingOccurrences(of: "\", with: "\\\"")
        .replacingOccurrences(of: "\", with: "\\\"")
    let queryString = "**=" + escapedString + "\"cwdt"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])
    query.foundItemsHandler =  {
        (items : [CSSearchableItem]) -> Void in
        /* process received items */
    }
    query.completionHandler = {  (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }
    query.start()
}
func search(userQuery: String) {
    query.cancel();
    let escapedString = userQuery.replacingOccurrences(of: "\\", with: "\\\\")
        .replacingOccurrences(of: "\"", with: "\\\"")
    let queryString = "\"=" + escapedString + "\"cwdt"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])

    query.foundItemsHandler = {
        (items: [CSSearchableItem]) -> Void in
        /* process received items */
    }

    query.completionHandler = { (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }

    query.start()
}
func search(userQuery : String) {
    query.cancel();
    let escapedString = userQuery.replacingOccurrences(of: "\\", with: "\\\\\\")
        .replacingOccurrences(of: '"', with: '\"')
    let queryString = "**=" + escapedString + "cwdt"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])

    query.foundItemsHandler = {
        (items : [CSSearchableItem]) -> Void in
        /* process received items */
    }

    query.completionHandler = { (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }

    query.start()
}
func search(userQuery : String) {
    query.cancel();
    let escapedString = userQuery.replacingOccurrences(of: "\\", with: "\\\\\")
        .replacingOccurrences(of: "\", with: "\\\\\\")
    let queryString = "**=" + escapedString + "\\cwdt"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])

    query.foundItemsHandler = {
        (items : [CSSearchableItem]) -> Void in
        /* process received items */
    }

    query.completionHandler = { (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }

    query.start()
}
```swift
func search(userQuery: String) {
    query.cancel();
    let escapedString = userQuery.replacingOccurrences(of: "\", with: "\\")
        .replacingOccurrences(of: "\"", with: "\\\"")
    let queryString = "**=" + escapedString + "cwdt"
    query = CSSearchQuery(queryString: queryString, attributes: ["displayName"])

    query.foundItemsHandler = {
        (items: [CSSearchableItem]) -> Void in
        /* process received items */
    }

    query.completionHandler = { (err) -> Void in
        /* finish processing */
        self.updateDisplay()
    }

    query.start()
}
```
Demo

New Core Spotlight Search API
Adopting Search APIs
Steps and best practices

Indexing content
Rich presentation
Launching into your app
CoreSpotlight Search API
Optimizing for ranking
Optimizing for Ranking

Vipul Ved Prakash Apple, Inc.
Ranking
Insight into Spotlight’s ranking algorithm
Ranking

Insight into Spotlight’s ranking algorithm

- Engagement Ratio
- Content Popularity
- Other Factors
Engagement Ratio
Historical app performance

- Device Side
- Per Query
- Global
Content Popularity
View and Link Counts

- URL Reputation
- NSUserActivity
- Universal Link Popularity
Universal Link Popularity

Yosemite National Park

0 0 1 0 0 0 0
Universal Link Popularity

Yosemite National Park

Frequency

Zion
Yosemite
Yellowstone
Bryce Canyon
Great Basin
Haleakala

Frequency
Best Practices for Ranking

Elements of Success

Follow CoreSpotlight best practices
Index Universal Links
Link NSUserActivities to Universal Links and CoreSpotlight items
Provide relevant content and rich presentation
Implement continue Search in App
Conclusion
Key Take-Aways

Spotlight is universal search on iOS
Customers expect well functioning apps to be searchable
If you have not implemented Search APIs, update today
New features for deeper search and increased engagement
<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Engineering Privacy for Your Users</td>
<td>Pacific Heights</td>
<td>Wednesday 4:00PM</td>
</tr>
<tr>
<td>Introducing SiriKit</td>
<td>Presidio</td>
<td>Wednesday 5:00PM</td>
</tr>
<tr>
<td>Increase Usage of Your App With Proactive Suggestions</td>
<td>Mission</td>
<td>Friday 1:40PM</td>
</tr>
<tr>
<td>Labs</td>
<td>Frameworks Lab A</td>
<td>Thursday 12:00PM</td>
</tr>
<tr>
<td>-------------------</td>
<td>------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>App Search and Spotlight Lab</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
More Information

https://search.developer.apple.com