What's New in Xcode 11

Ken Orr, Developer Tools
Demo
Workflow updates

Brendan Donohoe, Developer Tools
Demo
Workflow updates
High Contrast Themes
Code Drag and Drop Improvements
Minimap
Spell Checking
Enhanced Comment Rendering
Nested Code Folding
Mark Separator Rendering
Finer Grained Syntax Coloring Options
Inline Diff
Code Completion Improvements
Code Drag and Drop Improvements

Spell Checking

Enhanced Comment Rendering

Mark Separator Rendering

Finer Grained Syntax Coloring Options

Code Completion Improvements

High Contrast Themes

Nested Code Folding

Minimap
Code Completion Improvements

Spell Checking

Code Drag and Drop Improvements

High Contrast Themes

Minimap

Nested Code Folding

Mark Separator Rendering

Inline Diff

Finer Grained Syntax Coloring Options

Enhanced Comment Rendering

Minimap
Code Completion Improvements

Code Drag and Drop Improvements

High Contrast Themes

Spell Checking

Minimap

Nested Code Folding

Mark Separator Rendering

Inline Diff

Finer Grained Syntax Coloring Options

Enhanced Comment Rendering
warning("message")
var colors: [UIColor] = [.system]
var colors: [UIColor] = [.system]
var colors: [UIColor] = [.systemRed, .systemBlue, .systemGray, .systemMint, .systemPink]
struct TimeMachine {
    func travel(backTo: Date) {...}
    func travel(forwardTo: Date) {...}
}

let timeTravel = TimeMachine()
timeTravel.travel( )
struct TimeMachine {
    func travel(backTo: Date) {...}
    func travel(forwardTo: Date) {...}
}

let timeTravel = TimeMachine()
timeTravel.travel(□)
struct TimeMachine {
    func travel(backTo: Date) {...}
    func travel(forwardTo: Date) {...}
}

let timeTravel = TimeMachine()
timeTravel.travel((backTo: Date) (forwardTo: Date)
Swift Package Manager
Swift Package Manager

Introduced with Swift open source

Thousands of packages

Great way to factor your project
Swift Package Manager

Introduced with Swift open source

Thousands of packages

Great way to factor your project
Swift Package Manager

Packages are first class

Integrated throughout Xcode

Great for consuming and producing
Demo

Patrick Killian, Developer Tools
<table>
<thead>
<tr>
<th>Demo</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Adopting Swift Packages in Xcode</strong></td>
</tr>
<tr>
<td><strong>Creating Swift Packages</strong></td>
</tr>
<tr>
<td><strong>Binary Frameworks in Swift</strong></td>
</tr>
</tbody>
</table>
Source Control
// Copyright © 2019 Apple Inc. All rights reserved.

import Foundation

// Journal are completed Plans
typealias Journal = Plan

class Plan: Codable, Hashable {

    // Title of this plan
    let title: String

    // The region where the plan is for.
    let region: String

    // The events the user scheduled for this plan.
    var events: [Event] = []

    // Completed plans are listed under 'Journals', where they can be reviewed by the user.
    let completed: Bool

    init(title: String, region: String, completed: Bool = false) {
        self.title = title
        self.region = region
        self.completed = completed
    }

    static func ==(lhs: Plan, rhs: Plan) -> Bool {
        return lhs.title == rhs.title
    }

    func hash(into hasher: inout Hasher) {
    }
}
class Plan : Codable,

    // Title of this plan
    let title: String

    // The region where the plan is for.
    let region: String

    // The events the user scheduled for this plan.
    var events: [Event] = []

    // Completed plans are listed under 'Journals', where they can be reviewed by the user.
    let completed: Bool

    init(title: String, region: String, completed: Bool = false) {
        self.title = title
        self.region = region
        self.completed = completed
    }

    static func ==(lhs: Plan, rhs: Plan) -> Bool {
        return lhs.title == rhs.title
    }

    func hash(into hasher: inout Hasher) {
        // Hashing implementation
    }
import Foundation

class Plan : Codable, Hashable {
    // Title of this plan
    let title: String

    // The region where the plan is for.
    let region: String

    // The events the user scheduled for this plan.
    var events: [Event] = []

    let completed: Bool

    init(title: String, region: String, completed: Bool = false) {
        self.title = title
        self.region = region
        self.completed = completed
    }

    static func ==(lhs: Plan, rhs: Plan) -> Bool {
        return lhs.title == rhs.title
    }

    func hash(into hasher: inout Hasher) {
        // Implement hash function
    }
}
class Plan : Codable {
    // Title of the plan.
    let title: String

    // The region where the plan is for.
    let region: String

    // The events the user scheduled for this plan.
    var events: [Event] = []

    // Completed plans are listed under 'Journals', where they can be reviewed by the user.
    let completed: Bool

    init(title: String, region: String, completed: Bool = false) {
        self.title = title
        self.region = region
        self.completed = completed
    }

    static func ==(lhs: Plan, rhs: Plan) -> Bool {
        return lhs.title == rhs.title
    }

    func hash(into hasher: inout Hasher) {
    }
}
// Copyright © 2020. All rights reserved.

import Foundation

typealias Journal = Codable

class Plan : Codable {

// Title of the plan.
let title: String

// The region where the plan is for.
let region: String

// The events the user scheduled for this plan.
var events: [Event] = []

// Completed plans are listed under 'Journals', where they can be reviewed by the user.
let completed: Bool

init(title: String, region: String, completed: Bool = false) {
    self.title = title
    self.region = region
    self.completed = completed
}

static func ==(lhs: Plan, rhs: Plan) -> Bool {
    return lhs.title == rhs.title
}

func hash(into hasher: inout Hasher) {

First cut of "travel journal" feature.
import Foundation

class Plan : Codable, Hashable {

    // Title of this plan
    let title: String

    // The region where the plan is for.
    let region: String

    // The events the user scheduled for this plan.
    var events: [Event] = []

    // Completed plans are listed under 'Journals',
    // where they can be reviewed by the user.
    var completed: Bool

    init(title: String, region: String, completed: Bool = false) {
        self.title = title
        self.region = region
        self.completed = completed
    }

    static func ==(lhs: Plan, rhs: Plan) -> Bool {
        return lhs.title == rhs.title
    }

    func hash(into hasher: inout Hasher) {
        title.hash(into: &hasher)
    }
}
Add a few more points of interest.

Fix height of Journal table view cells.
<table>
<thead>
<tr>
<th>Commit Message</th>
<th>Identifier</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>9233a5d</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>8354308</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>20c37deab</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>42e41d4</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>92b27be</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>0d3308f</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>1f5f60f7</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>43ef7d4</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>2de8a8a</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>13b4ea2</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>b6f: [README] Update readme</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>c7bf4c</td>
<td>5/22/19</td>
</tr>
<tr>
<td></td>
<td>727b853</td>
<td>5/22/19</td>
</tr>
</tbody>
</table>

- Author: Edward Sanchez
- Description: Initial commit for a feature.
extension JournalCollectionViewController: UICollectionViewController, UICollectionViewDelegateFlowLayout {

  func viewDidLoad() {
    super.viewDidLoad()
  }

  // Navigation

  // Storyboard-based application, you will often want to do a little preparation before navigation
  func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    let cell = sender as? JournalCollectionViewCell else { return }
    detailController = segue.destination as? JournalDetailTableViewController {
      cell.journal
  }

  // UICollectionViewDataSource

  func numberOfSections(in collectionView: UICollectionView) -> Int {
    return 1
  }

useIdentifier = "JournalCell"

collectionView: UICollectionView, collectionViewDelegateFlowLayout { 

func viewDidLoad() { 

viewDidLoad() 

}

Storyboard-based application, you will often want to do a little preparation

func prepare(for segue: UIStoryboardSegue, sender: Any?) { 

let cell = sender as? JournalCollectionViewCell else { return }

detailController = segue.destination as? JournalDetailTableViewController { 

detailController.journal = cell.journal

UICollectionViewDataSource

func numberOfSections(in collectionView: UICollectionView) -> Int { 

}
Journal Detail
Prototype Cells

Golden Gate Bridge

*“Perfect wonder is the most beautiful bridge in the world.”*

Table View
Prototype Content
Journal Detail
Prototype Cells

Golden Gate Bridge

Table View
Prototype Content

Table View
Prototype Content
<table>
<thead>
<tr>
<th>Font Style</th>
<th>Small</th>
<th>Medium</th>
<th>Large</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ultralight</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Thin</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Light</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Regular</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Medium</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Semibold</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Bold</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Heavy</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
<tr>
<td>Black</td>
<td>✂️</td>
<td>✂️</td>
<td>✂️</td>
</tr>
</tbody>
</table>
Demo

Lisa Xiao, Developer Tools
<table>
<thead>
<tr>
<th>Demo</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Creating Great Localized Experiences with Xcode 11</td>
<td>Tuesday, 10:00AM</td>
</tr>
<tr>
<td>Introducing iPad Apps for Mac: Building Your First App</td>
<td>Tuesday, 2:00PM</td>
</tr>
<tr>
<td>Taking iPad Apps for Mac to the Next Level</td>
<td>Friday, 5:00PM</td>
</tr>
<tr>
<td>Demonstration Title</td>
<td>Date &amp; Time</td>
</tr>
<tr>
<td>--------------------------------------------------------</td>
<td>---------------</td>
</tr>
<tr>
<td>Debugging in Xcode 11</td>
<td>Thursday, 10:00AM</td>
</tr>
<tr>
<td>Designing for Adverse Network and Temperature Conditions</td>
<td>Friday, 4:20PM</td>
</tr>
</tbody>
</table>
## INSTALLED APPS

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Travel</td>
<td>1</td>
<td>com.apple.dt.Travel</td>
</tr>
</tbody>
</table>

## DEVICE CONDITIONS

<table>
<thead>
<tr>
<th>Condition</th>
<th>Profile</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>None</td>
</tr>
</tbody>
</table>

Select a condition.

Start
INSTALLED APPS

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Travel</td>
<td>1</td>
<td>com.apple.dt.Travel</td>
</tr>
</tbody>
</table>

DEVICES CONDITIONS

Condition: Thermal State
Profile: Fair

Starts as though under slightly elevated thermal state.
### INSTALLED APPS

<table>
<thead>
<tr>
<th>Name</th>
<th>Version</th>
<th>Identifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Travel</td>
<td>1</td>
<td>com.apple.dt.Travel</td>
</tr>
</tbody>
</table>

### DEVICE CONDITIONS

<table>
<thead>
<tr>
<th>Condition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Thermal State</td>
<td>Fair</td>
</tr>
</tbody>
</table>

The system behaves as though under slightly elevated thermal state

Start
Testing

Reusable across schemes
Define what tests to include
Specify multiple configurations

Testing in Xcode
Thursday, 11:00AM
Simulator
Efficient Drawing: 60 FPS

CPU Use: Up to 90% Less
Efficient Drawing
60 FPS

CPU Use
UP TO 90% LESS

Warm Boot
UP TO 2x FASTER

Getting the Most Out of Simulator
Friday, 9:00AM
Instruments
UP TO 10x FASTER

Metal System Trace
Analyzing Data

Getting Started with Instruments

Thursday, 9:00AM
Canvas Editor for Code
Library Views and Modifiers
On-Device Previews
Rich Preview API
New Action Popover Actions
Canvas Editor for Code
Preview Pinning
Preview Debugging
Instruments Template
Development Time Assets
Code and Canvas Editors
Code Hot Swapping
Rich Preview API
Preview Pinning
Drawing and Animation

Drawing Paths and Shapes

Users receive a badge whenever they visit a landmark in their list. Of course, for a user to receive a badge, you’ll need to create one. This tutorial takes you through the process of creating a badge by combining paths and shapes, which you then overlay with another shape that represents the location.

If you want to create multiple badges for different kinds of landmarks, try experimenting with the overlaid symbol, varying the amount of repetition, or changing the various angles and scales.

Follow the steps to build this project, or download the finished project to explore on your own.

30min
Estimated Time

Project files

Xcode 11 beta
Drawing and Animation

Drawing Paths and Shapes

Users receive a badge whenever they visit a landmark in their list. Of course, for a user to receive a badge, you’ll need to create one. This tutorial takes you through the process of creating a badge by combining paths and shapes, which you then overlay with another shape that represents the location.

If you want to create multiple badges for different kinds of landmarks, try experimenting with the overlaid symbol, varying the amount of repetition, or changing the various angles and scales.

Follow the steps to build this project, or download the finished project to explore on your own.

30min
Estimated Time

Project files

Xcode 11 beta
struct DiscoverSwiftUIView: UIViewRepresentable {

    func makeUIView(context: Context) -> DiscoverView {
        return DiscoverView(sceneController: GlobeSceneController())
    }

    func updateUIView(_ uiView: DiscoverView, context: Context) {
        // your logic here.
    }
}

struct DiscoverSwiftUIView: UViewRepresentable {

    func makeUIView(context: Context) -> DiscoverView {
        return DiscoverView(sceneController: GlobeSceneController())
    }

    func updateUIView(_ uiView: DiscoverView, context: Context) {
        // your logic here.
    }
}
<table>
<thead>
<tr>
<th>Event</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing SwiftUI: Building Your First App</td>
<td>Tuesday, 11:00AM</td>
</tr>
<tr>
<td>SwiftUI Essentials</td>
<td>Wednesday, 11:00AM</td>
</tr>
</tbody>
</table>
