Source Control Workflows in Xcode

Session 418

Eric Dudiak, Xcode Engineer
Create a new repository
Make and view changes
Host and share repositories
Resolve and avoid conflicts
Pull requests and forks
Create a new repository
Make and view changes
Host and share repositories
Resolve and avoid conflicts
Pull requests and forks
Create a new repository
Make and view changes
Host and share repositories
Resolve and avoid conflicts
Pull requests and forks
Create a new repository
Make and view changes
Host and share repositories
Resolve and avoid conflicts
Pull requests and forks
Create a new repository
Make and view changes
Host and share repositories
Resolve and avoid conflicts
Pull requests and forks
A New Project
New Project

My Project
New Project

My Project
New Project

My Project
New Project

.git

My Project
New Project

My Project
New Project

My Project
New Project

My Project

My Project
New Project

My Project

My Project
New Project

My Project
New Project

My Project

A Commit
New Project

My Project

4f29be29c24
New Project

My Project

4f29be292c24
New Project

My Project

4f29be292c24
New Project

My Project

4f29be292c24 34b593c56cb4
New Project

My Project

4f29be292c24  34b593c56cb4  b4c207ad33da
New Project

My Project

4f29be292c24  34b593c56cb4  b4c207ad33da  1d152f1160b3
Making Changes
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
public class Planet: Hashable {
    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects

public class Planet: Hashable {
    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
}
public class Planet: Hashable {

  let name: String
  let color: UIColor

  private(set) var satellites: [Moon] = []
  private(set) var nearbyObjects: [TransNeptunianObject] = []
  private(set) var orbitingStar: Star?

  public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
  }

  /// Add a satellite orbiting the planet
  public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
  }

  /// Add a nearby object
  ///
  /// - Parameter nearbyObject: The object to add
  public func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
  }

  /// Remove a nearby object
  ///
  /// - Parameter nearbyObject: The object to remove
  public func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else { return }
    nearbyObjects.remove(at: index)
  }

  /// Finds the nearest object among the nearby objects

}
```swift
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
```
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
Committing
public class Planet: Hashable {

    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
```swift
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// - Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// - Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
```
private(set) var nearbyObjects: [TransNeptunianObject] = []

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addsSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// - Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// - Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
public class Planet: Hashable {

    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor)
    { self.name = name
      self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
      satellite.parentPlanet = self
      moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
      nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
      nearbyObjects.removeAll { $0 == nearbyObject }
    }

    public class Moon: TransNeptunianObject {
        let name: String
        let color: UIColor
        private(set) var moons: [Moon] = []

        public init(name: String, color: UIColor)
        { self.name = name
          self.color = color
        }
    }

    public class TransNeptunianObject {
        let name: String
        let color: UIColor
        private(set) var moons: [Moon] = []

        public init(name: String, color: UIColor)
        { self.name = name
          self.color = color
        }
    }

    public class Star {
        let name: String
        let color: UIColor
        private(set) var moons: [Moon] = []

        public init(name: String, color: UIColor)
        { self.name = name
          self.color = color
        }
    }

    public class Planet {
        let name: String
        let color: UIColor
        private(set) var moons: [Moon] = []

        public init(name: String, color: UIColor)
        { self.name = name
          self.color = color
        }
    }
}

// Finds the nearest object among the nearby objects
```swift
public class Planet: Hashable {

    var name: String
    var color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    func remove(nearbyObject: TransNeptunianObject) {
        nearbyObjects.removeAll { $0 == nearbyObject }
    }

    /// Finds the nearest object among the nearby objects
    func findNearestObject() -> TransNeptunianObject? {
        // Implementation...
    }

    //MARK: - Extensions

    extension Planet {
        var title: String {
            return name
        }
    }
}
```
Viewing History
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects

```swift
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
```
```swift
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
```
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
```swift
public class Planet: Hashable {

    let title: String
    let color: UIColor

    private(set) var moons: [Moon] = []
    private(set) var nearbyObjects: [SmallPlanet] = []
    private(set) var parentStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    public func addNearbyObject(_ nearbyObject: SmallPlanet) {
        nearbyObjects.append(nearbyObject)
    }

    public func removeNearbyObject(_ nearbyObject: SmallPlanet) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    // Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    // Add a nearby object
    public func addNearbyObject(_ nearbyObject: SmallPlanet) {
        nearbyObjects.append(nearbyObject)
    }

    // Remove a nearby object
    public func removeNearbyObject(_ nearbyObject: SmallPlanet) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    // Example usage
    let moon = Moon(name: "Moon", color: .blue)
    let smallPlanet = SmallPlanet(name: "Small Planet", color: .green)
    let nearbyObject = nearbyObjects.first ?? SmallPlanet(name: "Nearby Object", color: .brown)
    self.moons.append(moon)
    self.nearbyObjects.append(nearbyObject)
    self.removeNearbyObject(nearbyObject)
}
```

```swift
let name: String
let color: UIColor

private(set) var satellites: [Moon] = []
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
public func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
public func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else { return }
    nearbyObjects.remove(at: index)
}
```

```swift
let name: String
let color: UIColor
private(set) var satellites: [Moon] = []
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// ~ Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// ~ Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}
```

```swift
let title: String
let colour: UIColor
private(set) var moons: [Moon] = []
private(set) var nearbyObjects: [SmallPlanet] = []
private(set) var parentStar: Star?

public init(name: String, colour: UIColor) {
    self.name = name
    self.colour = colour
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// ~ Parameter nearbyObject: The object to add
func add(nearbyObject: SmallPlanet) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// ~ Parameter nearbyObject: The object to remove
func remove(nearbyObject: SmallPlanet) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}
```
let name: String
let color: UIColor
private(set) var satellites: [Moon] = []
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// - Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// - Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

func add(nearbyObject: SmallPlanet) {
    nearbyObjects.append(nearbyObject)
}

func remove(nearbyObject: SmallPlanet) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}
```swift
let name: String
let color: UIColor
private(set) var satellites: [Moon] = []
private(set) var nearbyObjects:
    [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// ~ Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// ~ Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}
```
let name: String
let color: UIColor
private(set) var satellites: [Moon] = []
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// - Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// - Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects

Eric Dudiaik May 28, 2018
Ken Orr May 16, 2017
Ken Orr May 18, 2017
Ken Orr May 20, 2017
Ken Orr May 16, 2017
Russ Bishop May 17, 2017
Russ Bishop May 20, 2017
Russ Bishop May 20, 2017
Russ Bishop May 17, 2017
Ken Orr May 19, 2017
Russ Bishop May 20, 2017
Russ Bishop May 20, 2017
Russ Bishop May 20, 2017
Russ Bishop May 20, 2017
Russ Bishop May 20, 2017
let name: String
let color: UIColor
private(set) var satellites: [Moon] = []
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
///
/// - Parameter nearbyObject: The object to add
func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
///
/// - Parameter nearbyObject: The object to remove
func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
```swift
let name: String
let color: UIColor
private(set) var satellites: [Moon] = []
private(set) var nearbyObjects: [TransNeptunianObject] = []

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a satellite orbiting the planet

/// Add a nearby object

/// Remove a nearby object

/// - Parameter nearbyObject: The object to remove

guard let index = nearbyObjects.index(of: nearbyObject) else {
    return
}

nearbyObjects.remove(at: index)

/// Finds the nearest object among the nearby objects
```
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    ///
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }
}
public class Planet: Hashable {
    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    
}
public class Planet: Hashable {
    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }
public class Planet: Hashable {
    let name: String
    let color: UIColor

    private(set) var satellites: [TransNeptunianObject] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    ///
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func removeNearbyObject(transNeptunianObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }
GitHub and the New Source Control Workflows in Xcode 9

WWDC 2017
Hosting
Hosting

By default, only a local copy of history
Storing off-site makes for a good backup
Allows syncing between machines
Supports collaboration
Hosting

In addition to any default Git server

Enhanced support for popular services
- GitHub
- Bitbucket
- GitLab.com

Including self-hosted and enterprise versions
Hosting

In addition to any default Git server

Enhanced support for popular services

- GitHub
- Bitbucket
- GitLab.com

Including self-hosted and enterprise versions
Sign in to your Bitbucket Cloud account

Account: eric@dudak.com
Password: [redacted]

Click the add (+) button to create a new account
Security
Security

Transferring Git data can be secured

Two methods

• HTTPS
• SSH

SSH requires additional setup
Create SSH key

Enter a passphrase for the new SSH key:

***************

Clone Using:
- HTTPS
- SSH

New repositories will be cloned from Bitbucket Cloud using SSH.

SSH Key: None

SSH key not configured

Create
<table>
<thead>
<tr>
<th>Folder Name</th>
<th>Last Updated</th>
<th>Owner</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCMStickers</td>
<td>Jul 7, 2017, 2:38 PM</td>
<td>dslask</td>
</tr>
<tr>
<td>swift-package-manager</td>
<td>Feb 22, 2017, 1:56 PM</td>
<td>dslask</td>
</tr>
<tr>
<td>TOTP</td>
<td>May 30, 2017, 1:28 PM</td>
<td>dslask</td>
</tr>
<tr>
<td>AsthmaHealth</td>
<td>Oct 26, 2016, 12:26 PM</td>
<td>ResearchGS</td>
</tr>
<tr>
<td>git</td>
<td>Today, 9:47 AM</td>
<td>git</td>
</tr>
<tr>
<td>GnuCSuccess</td>
<td>Jul 29, 2015, 11:24 AM</td>
<td>ResearchGS</td>
</tr>
<tr>
<td>ibag2x</td>
<td>May 26, 2016, 13:26 PM</td>
<td>ResearchGS</td>
</tr>
<tr>
<td>MyHeartCounts</td>
<td>Oct 27, 2015, 4:16 PM</td>
<td>ResearchGS</td>
</tr>
<tr>
<td>Nanodolo</td>
<td>Nov 25, 2017, 10:53 AM</td>
<td>toluco</td>
</tr>
<tr>
<td>ResearchKX</td>
<td>Apr 12, 2018, 9:19 PM</td>
<td>ResearchGS</td>
</tr>
<tr>
<td>swift</td>
<td>Today, 9:05 PM</td>
<td>apple</td>
</tr>
<tr>
<td>util</td>
<td>Yesterday, 9:07 AM</td>
<td>master</td>
</tr>
</tbody>
</table>
GitHub and the New Source Control Workflows in Xcode 9

WWDC 2017
Pulling and Pushing
public class Planet: Hashable {

    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects

```swift
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
///
/// - Parameter nearbyObject: The object to add
public func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
///
/// - Parameter nearbyObject: The object to remove
public func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
```
```swift
private(set) var nearbyObjects: [TransNeptunianObject] = []

private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
/// - Parameter nearbyObject: The object to add
public func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
/// - Parameter nearbyObject: The object to remove
public func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else { return }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
```
public class Planet {
    let name: String
    let color: UIColor

    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    ///
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    ///
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
///
/// - Parameter nearbyObject: The object to add
public func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
///
/// - Parameter nearbyObject: The object to remove
public func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
private(set) var nearbyObjects: [TransNeptunianObject] = []
private(set) var orbitingStar: Star?

public init(name: String, color: UIColor) {
    self.name = name
    self.color = color
}

/// Add a satellite orbiting the planet
public func addSatellite(_ satellite: Moon) {
    satellite.parentPlanet = self
    moons.append(satellite)
}

/// Add a nearby object
///
/// - Parameter nearbyObject: The object to add
public func add(nearbyObject: TransNeptunianObject) {
    nearbyObjects.append(nearbyObject)
}

/// Remove a nearby object
///
/// - Parameter nearbyObject: The object to remove
public func remove(nearbyObject: TransNeptunianObject) {
    guard let index = nearbyObjects.index(of: nearbyObject) else {
        return
    }
    nearbyObjects.remove(at: index)
}

/// Finds the nearest object among the nearby objects
public class Planet {

    let name: String
    let color: UIColor
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
}
Conflicts
```swift
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    func add(nearbyObject: SmallPlanet) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: SmallPlanet) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }
```

```swift
public class Planet: Hashable {

    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    // Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    // Add a nearby object
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    // Remove a nearby object
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    public class Planet: Hashable {

    let title: String
    let colour: UIColor
    private(set) var moons: [Moon] = []
    private(set) var nearbyObjects: [SmallPlanet] = []
    private(set) var parentStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    // Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    // Add a nearby object
    public func add(nearbyObject: SmallPlanet) {
        nearbyObjects.append(nearbyObject)
    }

    // Remove a nearby object
    public func remove(nearbyObject: SmallPlanet) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }
```

```swift
public class Planet: Hashable {

    let name: String
    let color: UIColor
    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    addSatellite(Moon())

    /// Add a nearby object
    public func addNearbyObject(_ nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    removeNearbyObject(TransNeptunianObject())

    /// - Parameter nearbyObject: The object to remove
    public func removeNearbyObject(_ nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }

    /// - Parameter nearbyObject: The object to remove
    public func removeNearbyObject(_ nearbyObject: SmallPlanet) {
        guard let index = nearbyObjects.index(of: nearbyObject) else { return }
        nearbyObjects.remove(at: index)
    }
```
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    /// - Parameter nearbyObject: The object to add
    public func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    /// - Parameter nearbyObject: The object to remove
    public func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
public class Planet: Hashable {

    let name: String
    let color: UIColor

    private(set) var satellites: [Moon] = []
    private(set) var nearbyObjects: [TransNeptunianObject] = []
    private(set) var orbitingStar: Star?

    public init(name: String, color: UIColor) {
        self.name = name
        self.color = color
    }

    /// Add a satellite orbiting the planet
    public func addSatellite(_ satellite: Moon) {
        satellite.parentPlanet = self
        moons.append(satellite)
    }

    /// Add a nearby object
    ///
    /// - Parameter nearbyObject: The object to add
    ///
    func add(nearbyObject: TransNeptunianObject) {
        nearbyObjects.append(nearbyObject)
    }

    /// Remove a nearby object
    ///
    /// - Parameter nearbyObject: The object to remove
    func remove(nearbyObject: TransNeptunianObject) {
        guard let index = nearbyObjects.index(of: nearbyObject) else {
            return
        }
        nearbyObjects.remove(at: index)
    }

    /// Finds the nearest object among the nearby objects
}
Pull Requests and Forks
Pull Request

A form of code review on a hosted service

Used when changes are done off of the main branch

The branch can be checked out locally to build and run
A form of code review on a hosted service

Used when changes are done off of the main branch

The branch can be checked out locally to build and run
A form of code review on a hosted service

Used when changes are done off of the main branch

The branch can be checked out locally to build and run
Pull Request

A form of code review on a hosted service

Used when changes are done off of the main branch

The branch can be checked out locally to build and run
Forks

Additional server copy of a repository
Has its own branches
Use PRs to re-integrate work
Clone separately or as second remote
Forks

Additional server copy of a repository
Has its own branches
Use PRs to re-integrate work
Clone separately or as second remote
Summary

Using Git locally
Viewing history
Syncronizing changes
Resolving conflicts
Hosting features
More Information

https://developer.apple.com/session418

GitHub and the New Source Control Workflows in Xcode 9

WWDC 2017

Xcode Open Hours

Tools Lab B

Tuesday–Thursday