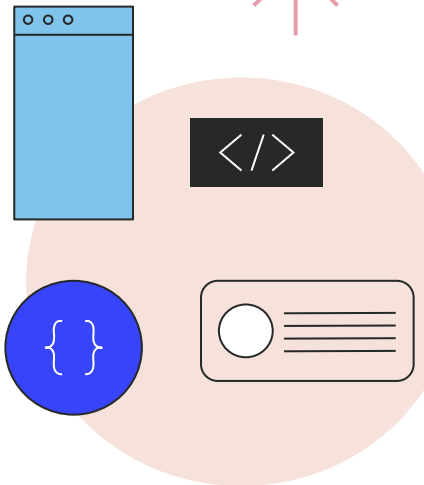
iOS Development at Mercari

Maintaining great build speeds on Local and CI environments

Tips to make use of SwiftUI in production



mercari

Speakers

Manoj Kumar Gubba

Sachin Nautiyal

Archita Bansal

mercari

Some Stats About Mercari

- ~ 50 active iOS developers
- Mono repo for ~3 iOS apps
- 120+ screens
- 550+ modules
- 100% Swift and SwiftUI*

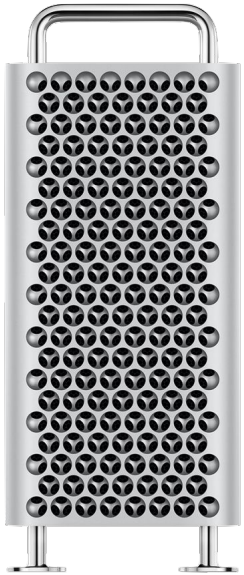
mercari

How to maintain good incremental build speeds?

mercari

How to maintain good incremental build speeds?

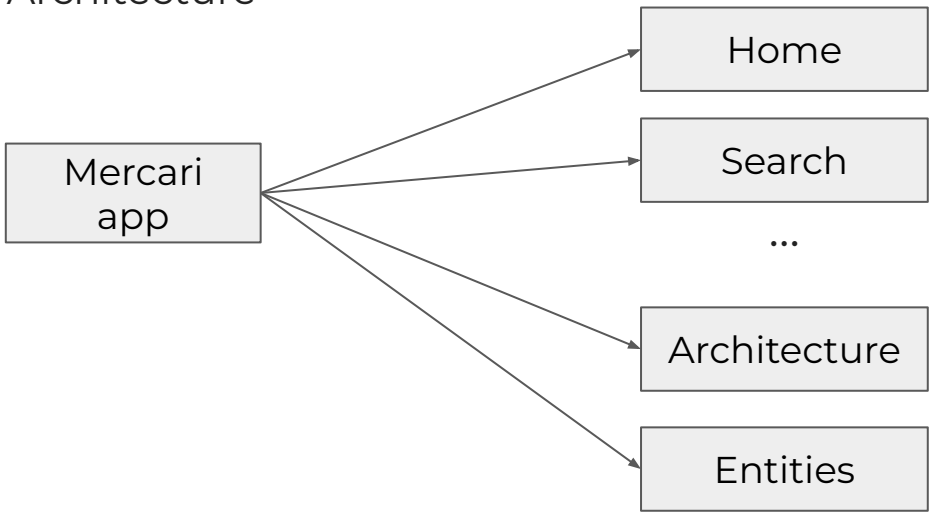
- Use powerful devices 😁



mercari

How to maintain good incremental build speeds ?

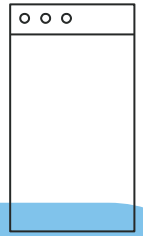
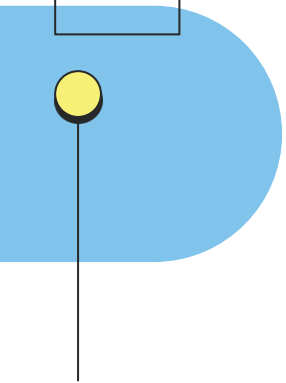
- Using a Modular Architecture



- But when we pull the changes, a lot of modules get changed, which doesn't help much with the build speed optimizations made by splitting app into small modules.

mercari



Solution ?

Use Build Cache

mercari

How to cache the builds ?

- Xcode Build Systems doesn’t have support for build cache.
- Need to choose alternatives:
  - Bazel
  - Buck
  - Others
- And we chose Bazel.

mercari

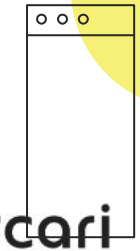
What is Bazel ?

Bazel is a build system developed by Google.

mercari

Why Bazel ?

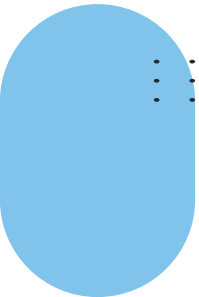
- open source.
- has good support for iOS.
- fast, correct, and extensible build tool.



mercari

Benefits received from Bazel

- Incremental build speeds with no changes were ~3 seconds.
- Build speeds after pulling changes involving several modules takes ~1 minute.
- Creating an .ipa file takes ~5 minutes.
- Unit tests for the complete app take ~10 minutes.



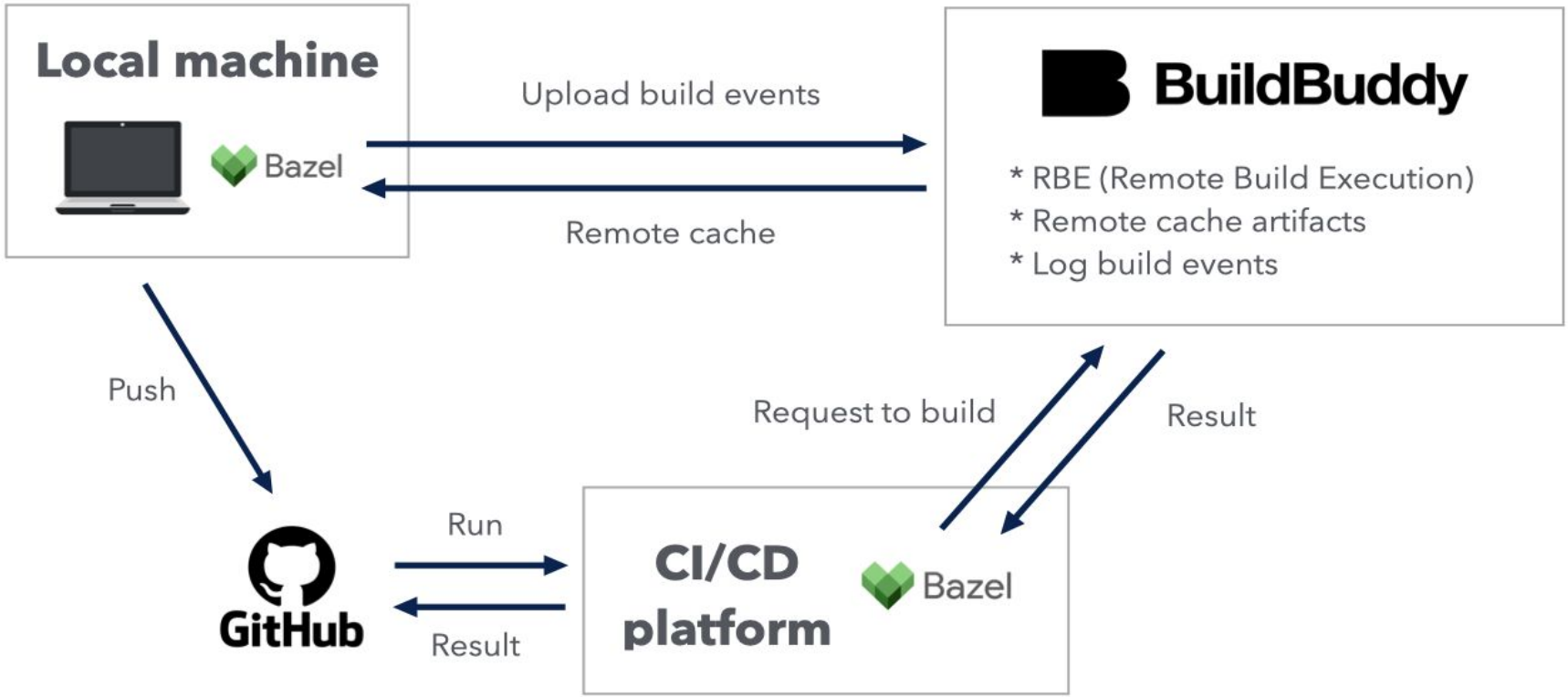
mercari

Clarifying some misconceptions about using Bazel

- We will still be using Xcode even after migrating to Bazel.
- Building apps, running UI & Unit testing can still be done directly from the Xcode UI.
- The syntax highlighting and jump to definition features of Xcode will still work.
- Dependency managers like Cocoapods, SPM, Carthage can still be used along with bazel.
- There might be a bit of overhead for first build when using Bazel, but incremental builds will be much faster.

mercari

How we cache the builds ?



Bazel demo

- Walkthrough the bazel demo project
- Before/after folder contents
- Run the sample app
- Create Xcode project
- Adding a new module to the app
- Use Remote Cache.

mercari

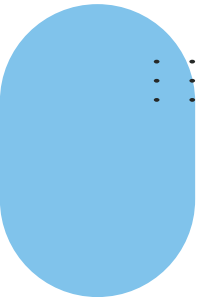
How does Bazel work ?

Workspace

- Used for defining the dependencies.
- Denotes the root folder of the repo.

Build

- Bazel uses build file using dependencies from workspace to build the app.
- Resources, code files everything is linked via build files.



mercari

Bazel cache in action

Cache stats

Action cache (AC)
Maps action hashes to action result metadata

- 4 hits
- 0 misses

Content addressable store (CAS)
Stores files (including logs & profiling info)

- 13 hits
- 1 writes

Volume
Total size of files uploaded & downloaded from both caches

- 153.4KB downloaded
- 6.839KB uploaded

Throughput
Upload / download speed (time-weighted avg)

- 24.37 Mbps download
- 17.52 Mbps upload

CPU savings
Cached CPU and actual CPU used

- 1.49s not executed
- 0s executed

Cache miss reasons
Reasons why actions missed cache

- 7 Different files (88%)
- 1 Unconditional execution (13%)

Action types
Number of actions with each action mnemonic

- 2 Symlink (25%)
- 1 CppArchive (13%)
- 1 SwiftCompile (13%)
- 1 BundleTreeApp (13%)
- 1 BazelWorkspaceStatusAction (13%)
- 2 Other (25%)

Runner types
Number of actions with each runner type

- 4 remote cache hit (50%)
- 3 internal (38%)
- 1 local (13%)

[Test run on BuildBuddy]

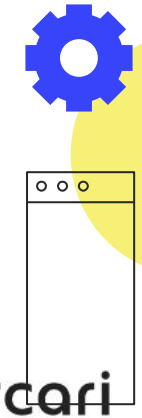
mercari

References

- About Bazel: https://bazel.build/about
- Mercan Blog: Fast and reliable iOS builds with Bazel at Mercari.
- Bazel Tutorial: Build an iOS App
- Remote Cache: BuildBuddy.io

mercari

Questions?



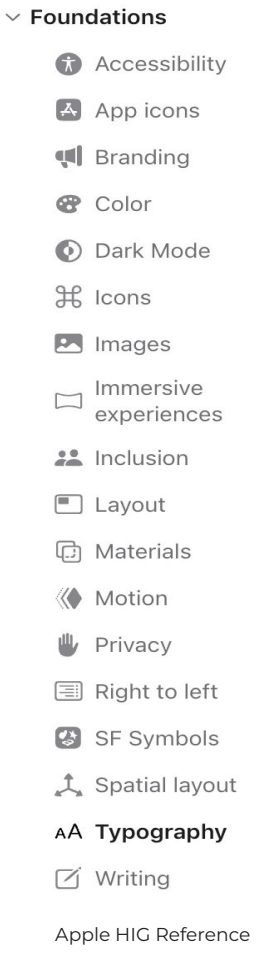
How to make use of SwiftUI in production

- Architecture
  - Custom architecture based on TCA and Clean
- Design System
- Solving Navigation
- Snapshot testing
  - Playbook

mercari

Design System

- Design System is a collection of assets and patterns that can be used to create consistent, high-quality user interfaces.
- Benefits
  - Ensure branding
  - Support accessibility
  - Dark Mode
- Contents
  - Colors
  - Icons
  - Typography
  - ...
  - Components
    - Reusable UI elements



mercari

SwiftUI Navigation

- SwiftUI has good APIs for handling navigation.
  - NavigationLink
  - navigationDestination (iOS 16+)

- But, there are several edge cases where they don't work as expected.

mercari

How we solved the SwiftUI navigation at Mercari ?

- By making use of UIKit Navigation APIs.
- Found that SwiftUI's NavigationView is internally using `UINavigationController` which made it easy to use the UIKit APIs.

mercari

Router

```
final class Router {
    weak var controller: UIViewController?

    func push<V: View>(_ view: V) {
        let next = UIHostingController(rootView: view)
        controller?.navigationController?.pushViewController(next, animated: true)
    }

    func pop() {
        controller?.navigationController?.popViewController(animated: true)
    }
}
```

mercari

How do we get the controller?

```
private struct HostController: UIViewControllerRepresentable {
    var router: Router

    func makeUIViewController(context: Context) -> UIViewController {
        UIViewController()
    }

    func updateUIViewController(_ uiViewController: UIViewController, context: Context) {
        router.controller = uiViewController
    }
}
```

mercari

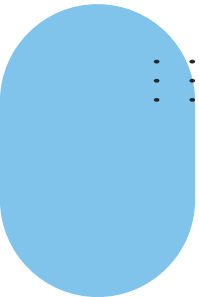
Router Helper

```
struct RouterProvider<Content: View>: View {
    let content: (Router) -> Content
    private let router = Router()

    var body: some View {
        content(router)
            .background(HostController(router: router))
    }
}
```

mercari

Navigation in Action



mercari

Snapshot Testing

- Fast changing SwiftUI APIs can be tested on production with these tests.
- We use playbook-ios open source library to write scenarios.
- Can be automated on CI
- Separate UI app for all UI components.

mercari