

How **Apple Intelligence** infuses and distills its models into the app with help from RAG

Etienne Vautherin - Swift Connection - September 23-24, 2024

Baby apps with software Developers

Practice technologies

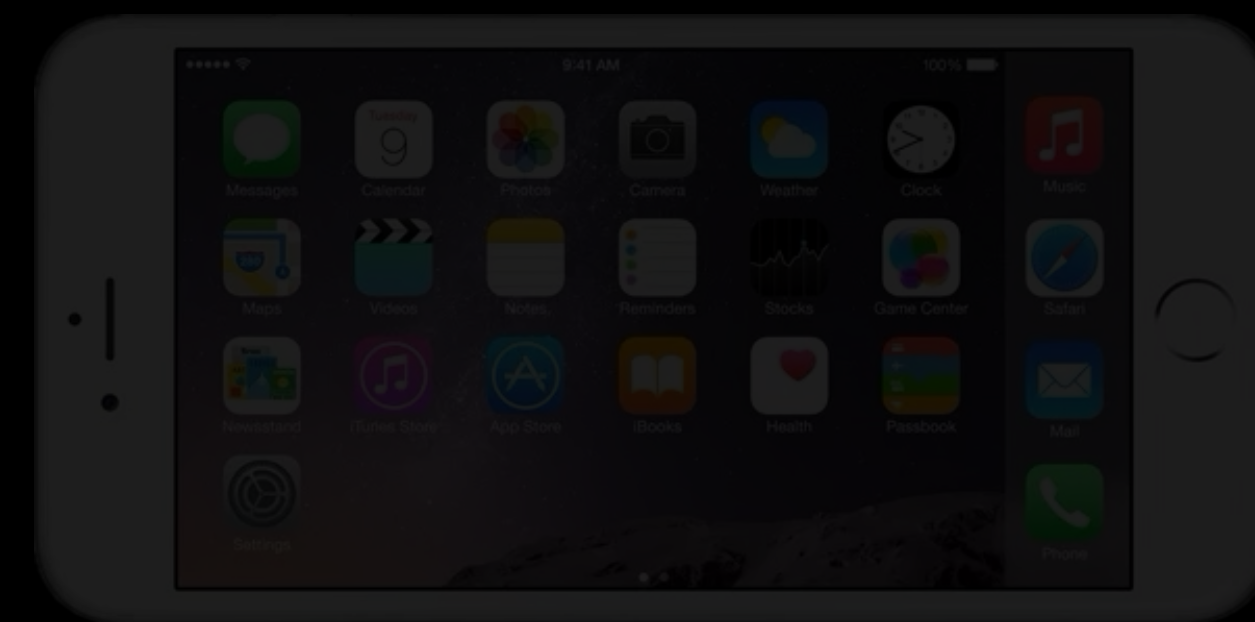
Sample Code (training)

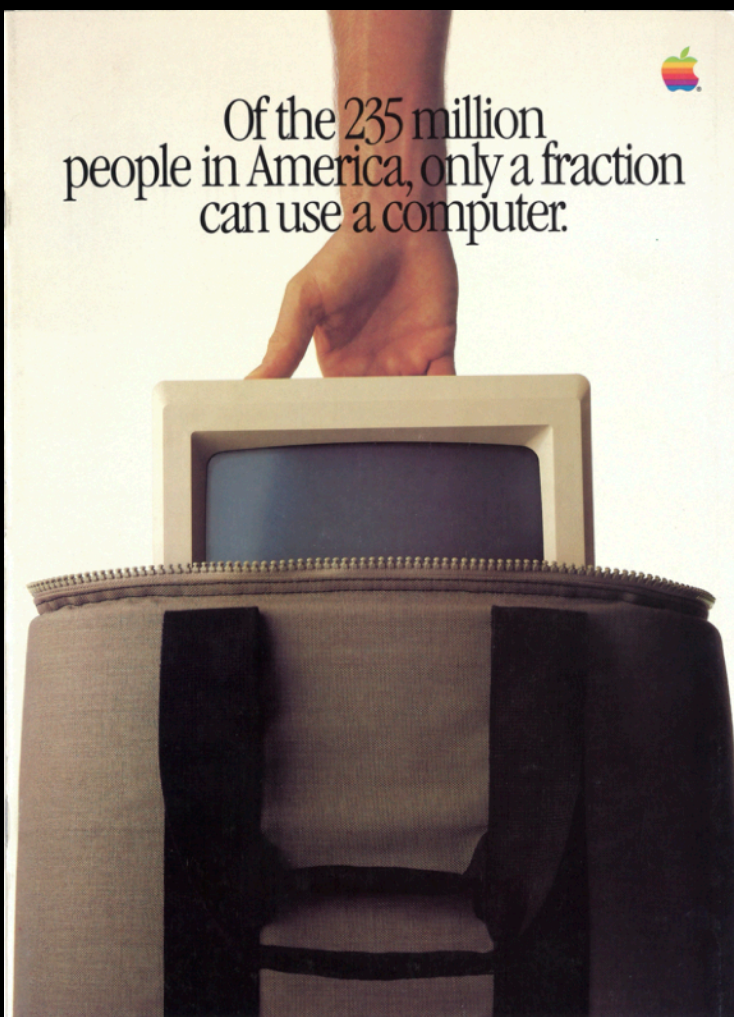
Test technologies

Proof Of Concept

Adopt technologies

Minimum Viable Product





12 years
Apple Developer Relations



15 years
Orange Innovation



A large flock of birds is flying in a circular pattern in the sky, creating a large, dark, circular shape. The background is a sunset landscape with a body of water reflecting the orange and yellow light of the sun. The sky is a mix of blue, purple, and orange.

Practice technologies
Sample Code (training)

Test technologies
Proof Of Concept

Adopt technologies
Minimum Viable Product

Practice technologies

Sample Code (training)

Test technologies

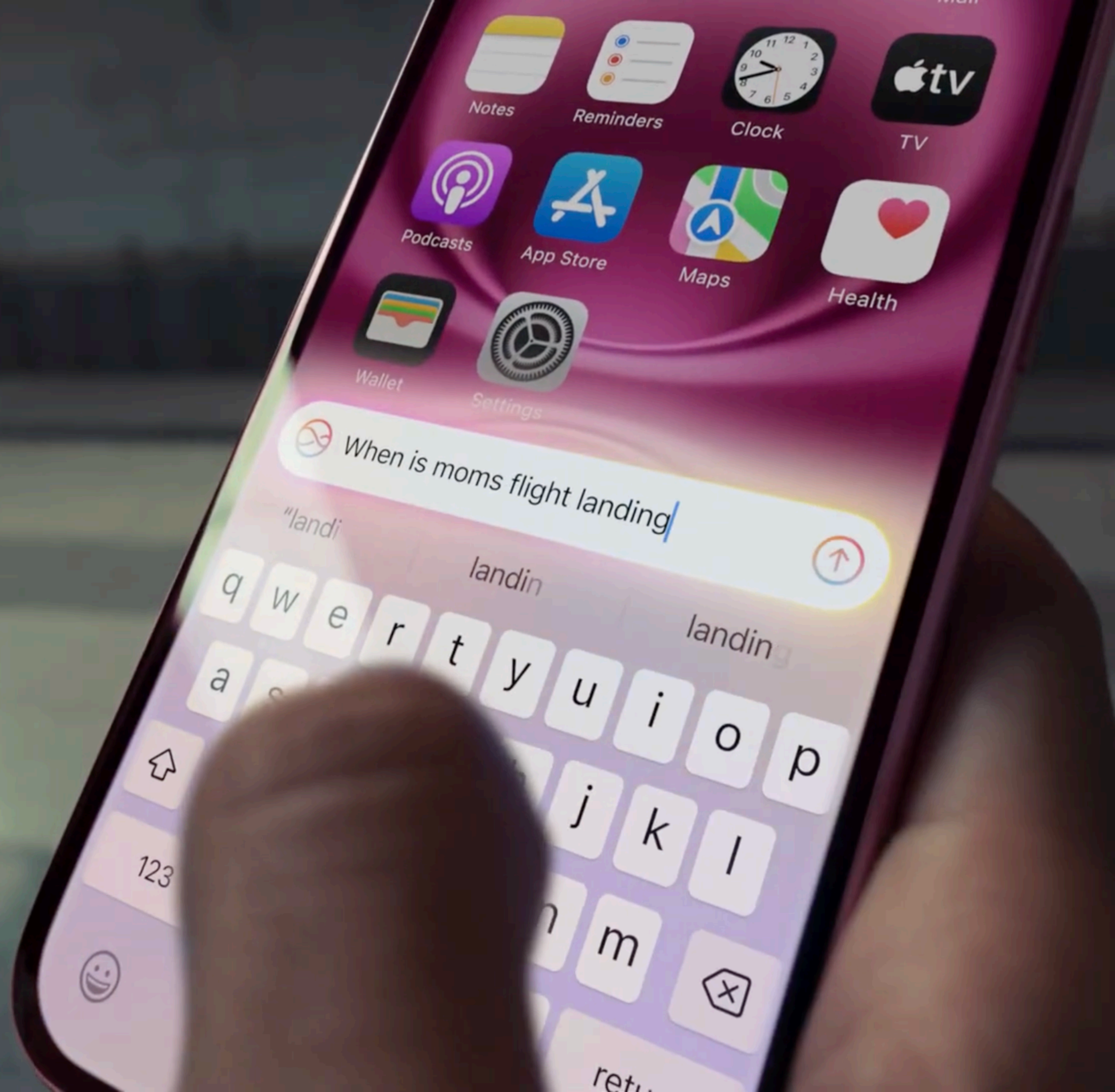
Proof Of Concept

Adopt technologies

Minimum Viable Product



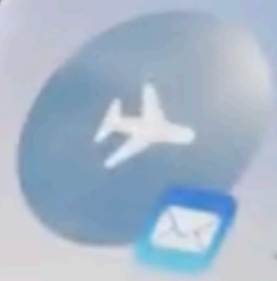
Crédit d'Impôt Recherche



9:41

5G

Your mom's flight lands at 11:18 PM.



UA3328, United Airlines **En Route**

✈ Mon, 8:45 PM

✈ Mon, 11:18 PM

ORD Term 1
SFO Term 3

Notes

Reminders

Clock

TV

Podcasts

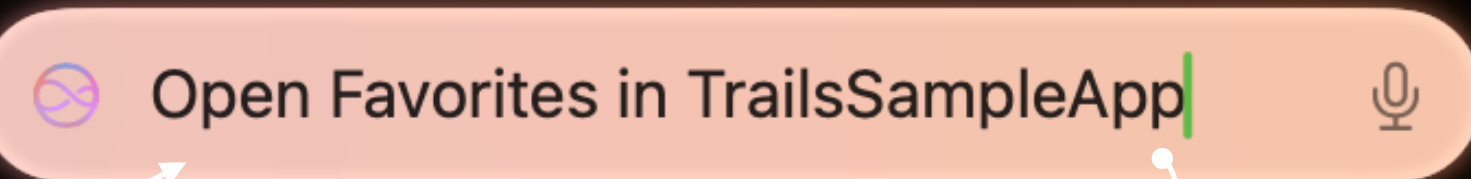
App Store

Maps

Health

Wallet

Today



Designated app
in a preconfigured phrase

Apple Intelligence

Open Favorites in TrailsSampleApp



When is moms flight landing



user can compose
functionality and elements
without any app reference

Apple Intelligence

Open Favorites in TrailsSampleApp



When is moms flight landing



App developer exposes
functionality and elements
that user can compose

How does it work?

Apple Domains

Photos and Videos

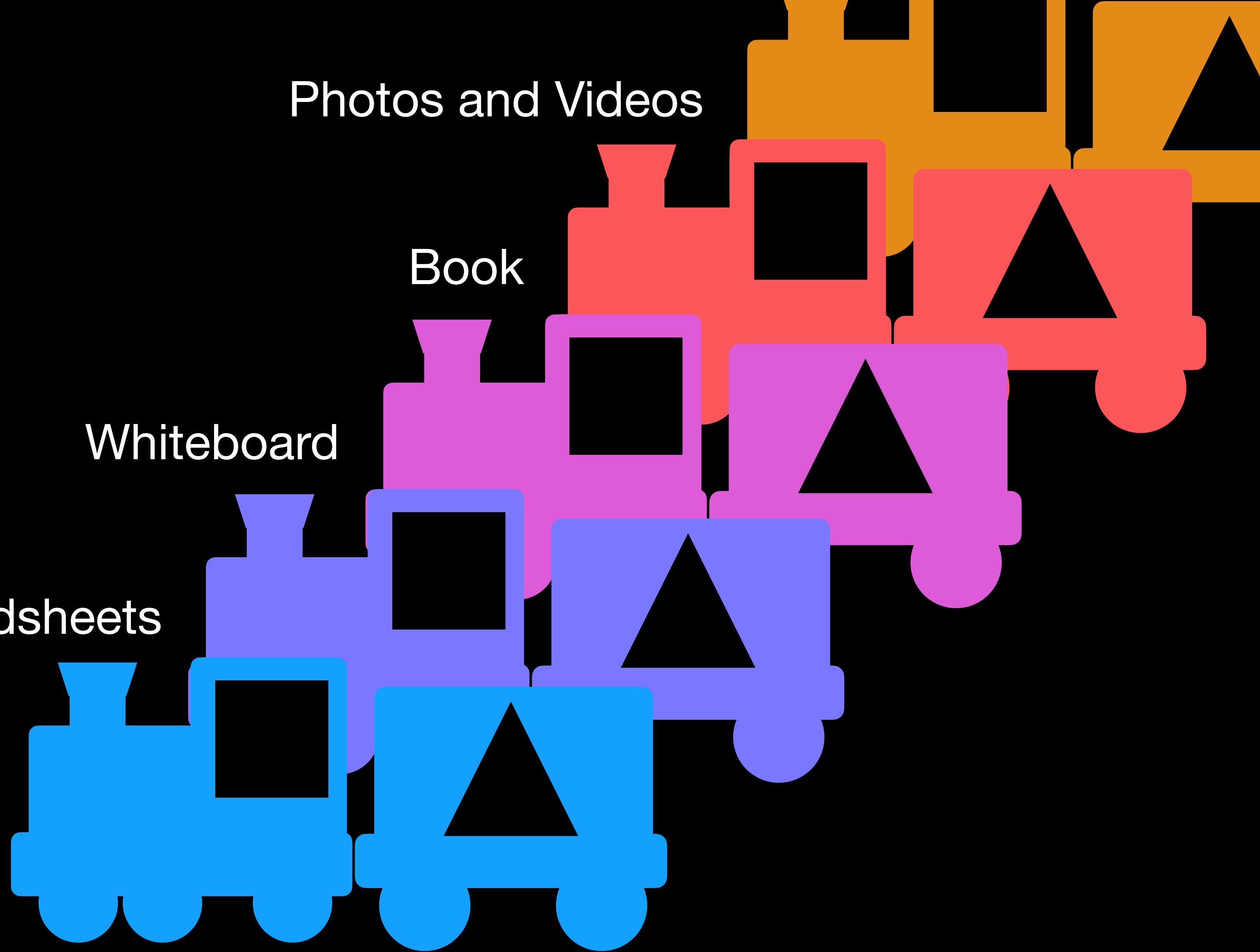
Book

Whiteboard

Spreadsheets

Email

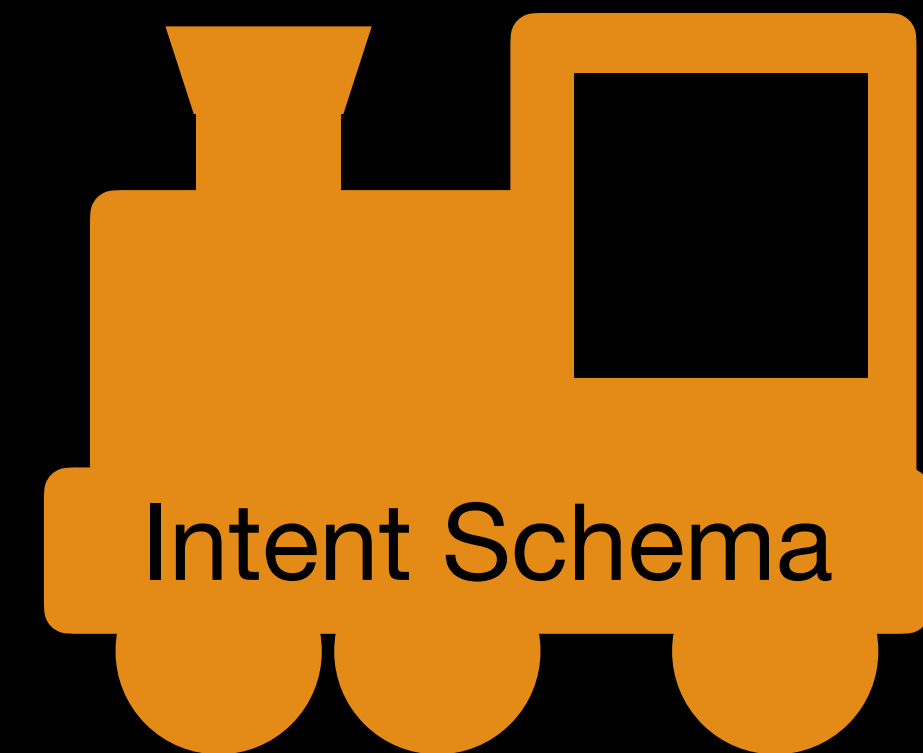
Etc.




```
@AssistantIntent(schema: .photos.openAsset)
struct OpenBabyIntent: OpenIntent {
    var target: BabyEntity
    ...
    @MainActor
    func perform() async throws -> some IntentResult {
        ...
    }
}
```

```
@AssistantEntity(schema: .photos.asset)
struct BabyEntity: IndexedEntity {
    ...
    static let defaultQuery = AssetQuery()

    let id: String
    ...
}
```



Intent Schema

.photos.openAsset



Entity Schema

.photos.asset


```
@AssistantIntent(schema: .mail.sendDraft)
struct SendDraftIntent: AppIntent {
    var target: MailDraftEntity
    var sendLaterDate: Date?

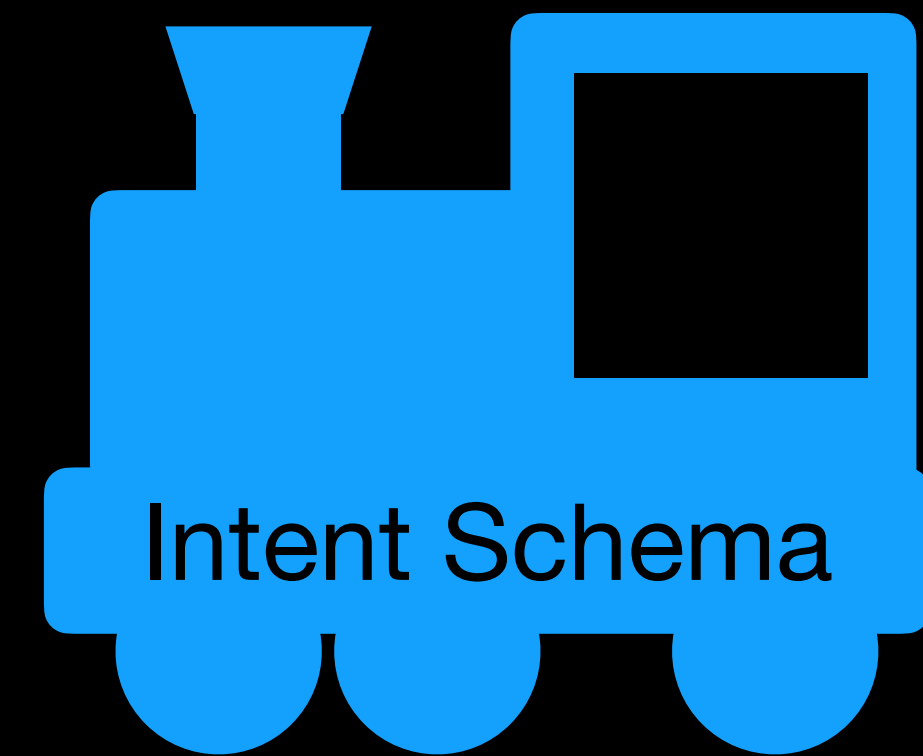
    @MainActor
    func perform() async throws -> some IntentResult {
        return .result()
    }
}
```

```
@AssistantEntity(schema: .mail.draft)
struct MailDraftEntity {

    static var defaultQuery = Query()

    struct Query: EntityStringQuery {
        ...
    }

    let id = UUID()
}
```

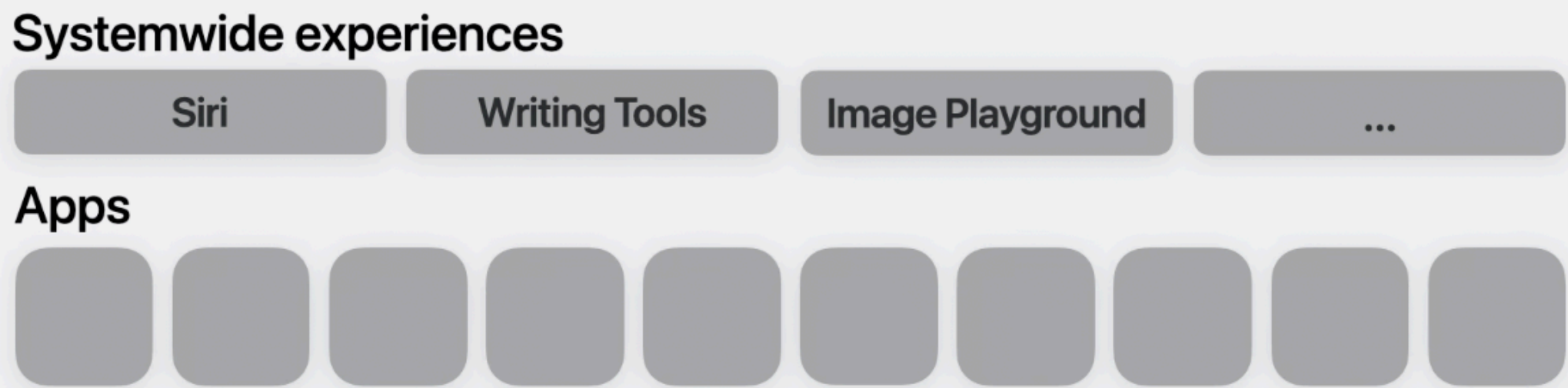


`.mail.sendDraft`

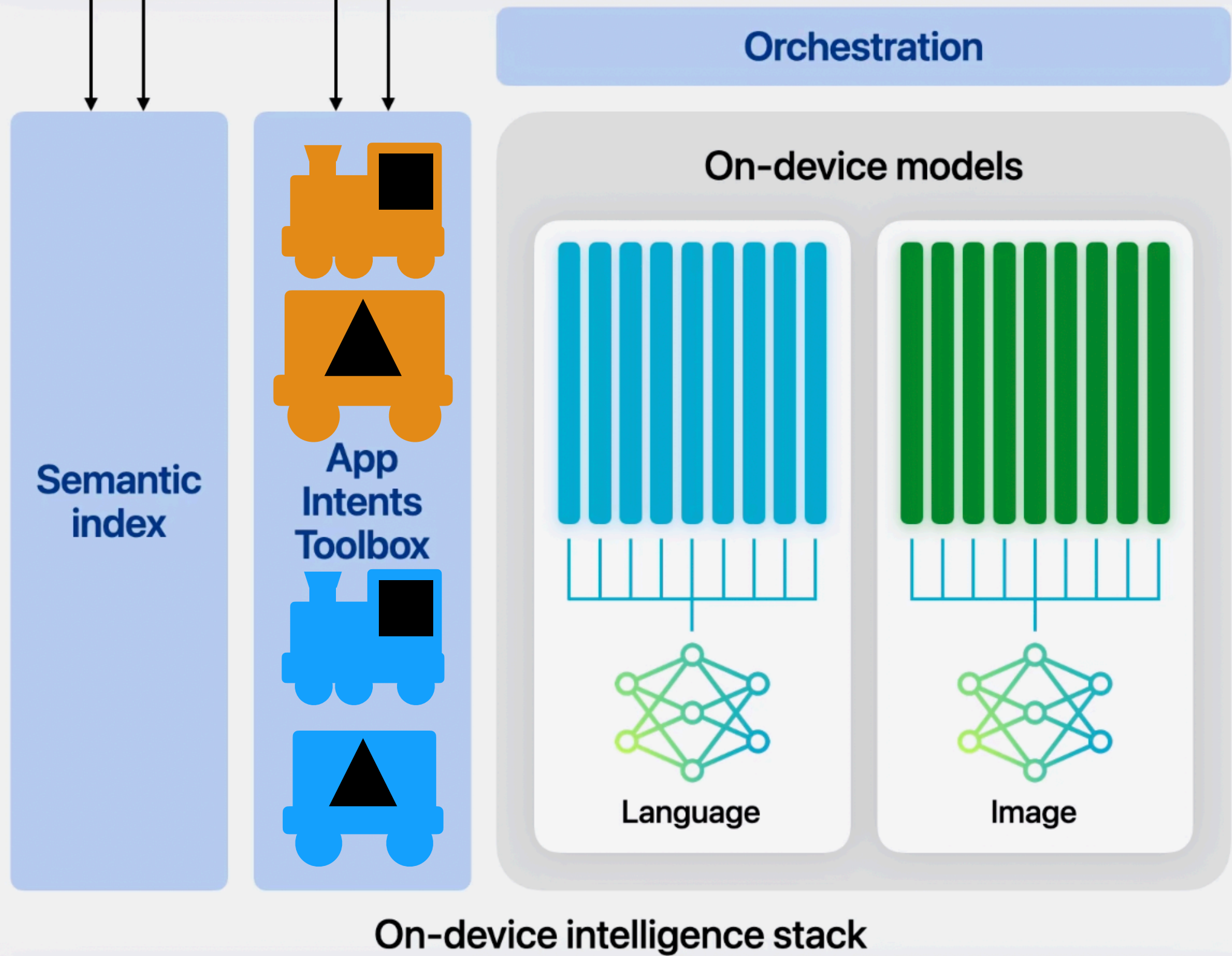


`.mail.draft`

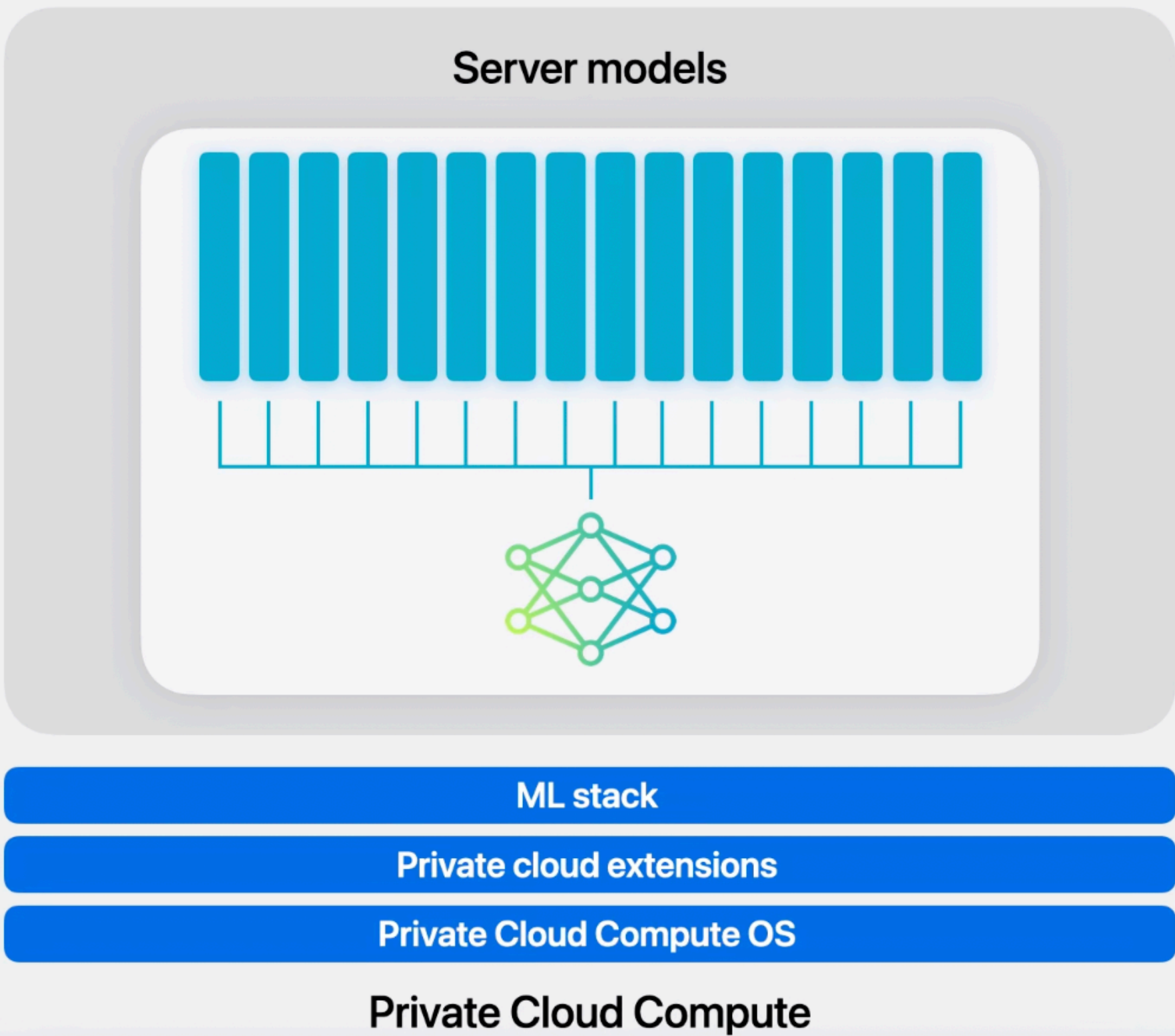
Apps and experiences



Personal Intelligence System



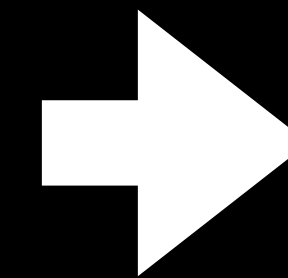
Apple silicon



"Conceptual" Intent



Comparison



Generated Intent

Transformation



Show the last photo of Mom with a red
dress



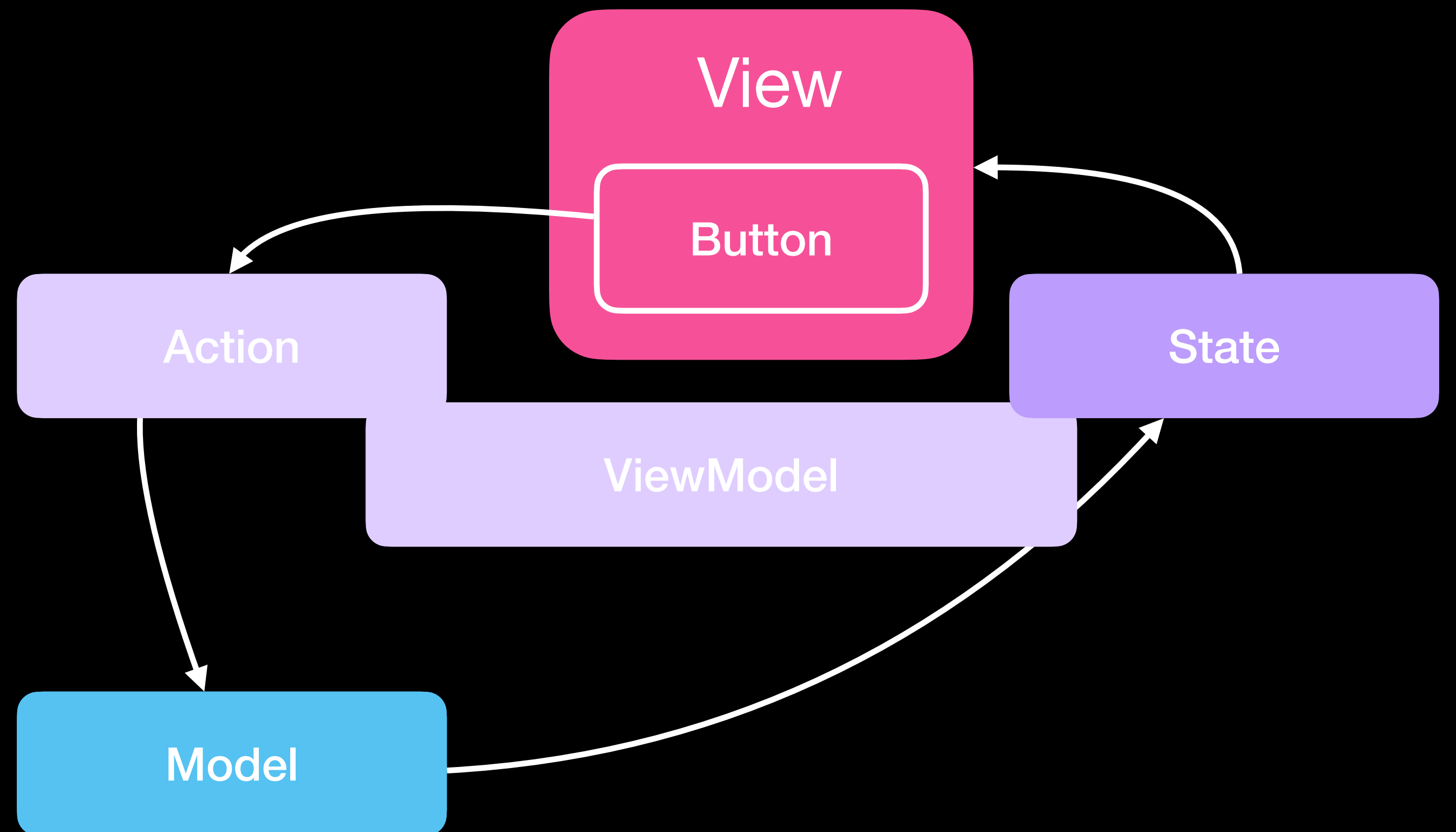
MVVM

View

ViewModel

Model

MVVM




```

struct BabyApp: App {
  let model: Model

  init() {
    let model = Model()
    self.model = model
  }

  var body: some Scene {
    WindowGroup {
      ContentView()
    }
    .environment(model)
  }
}

```

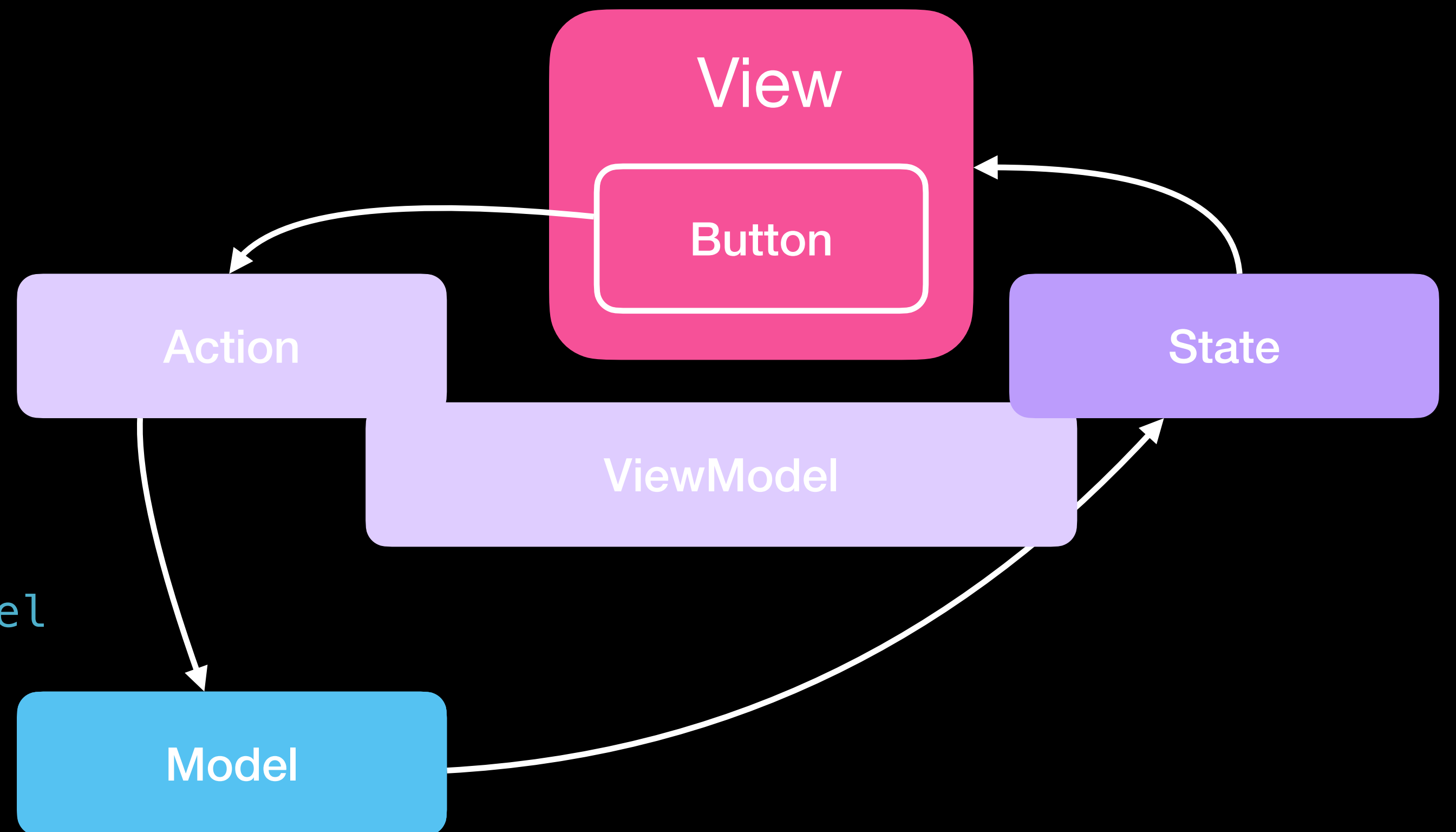
```

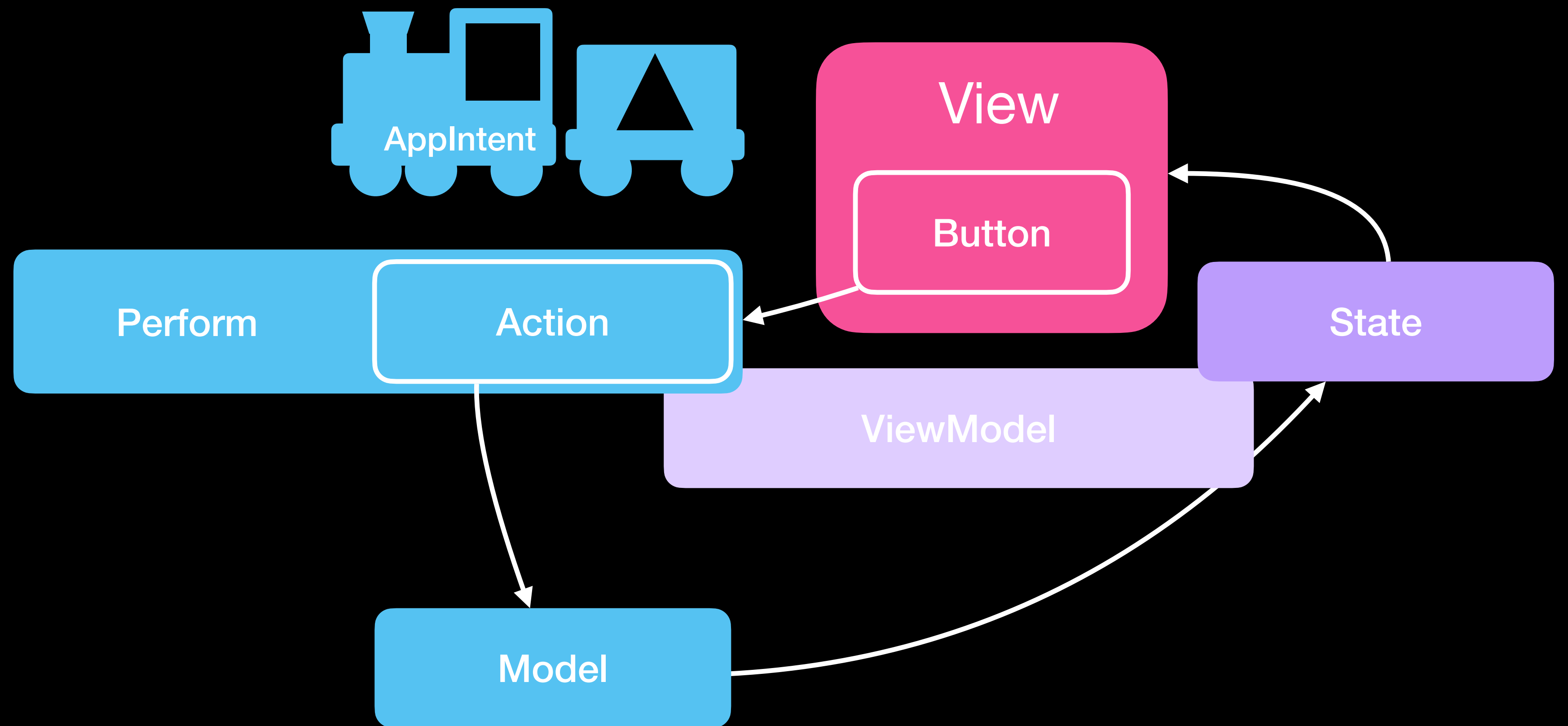
struct BabyView: View {
  @Environment(Model.self) private var model

  var body: some View {
    ...
  }
}

```

MVVM





```

struct BabyApp: App {
    let model: Model

    init() {
        let library = Model()
        AppDependencyManager.shared.add(dependency: library)
        self.model = model
    }

    var body: some Scene {
        WindowGroup {
            ContentView()
        }
        .environment(model)
    }
}

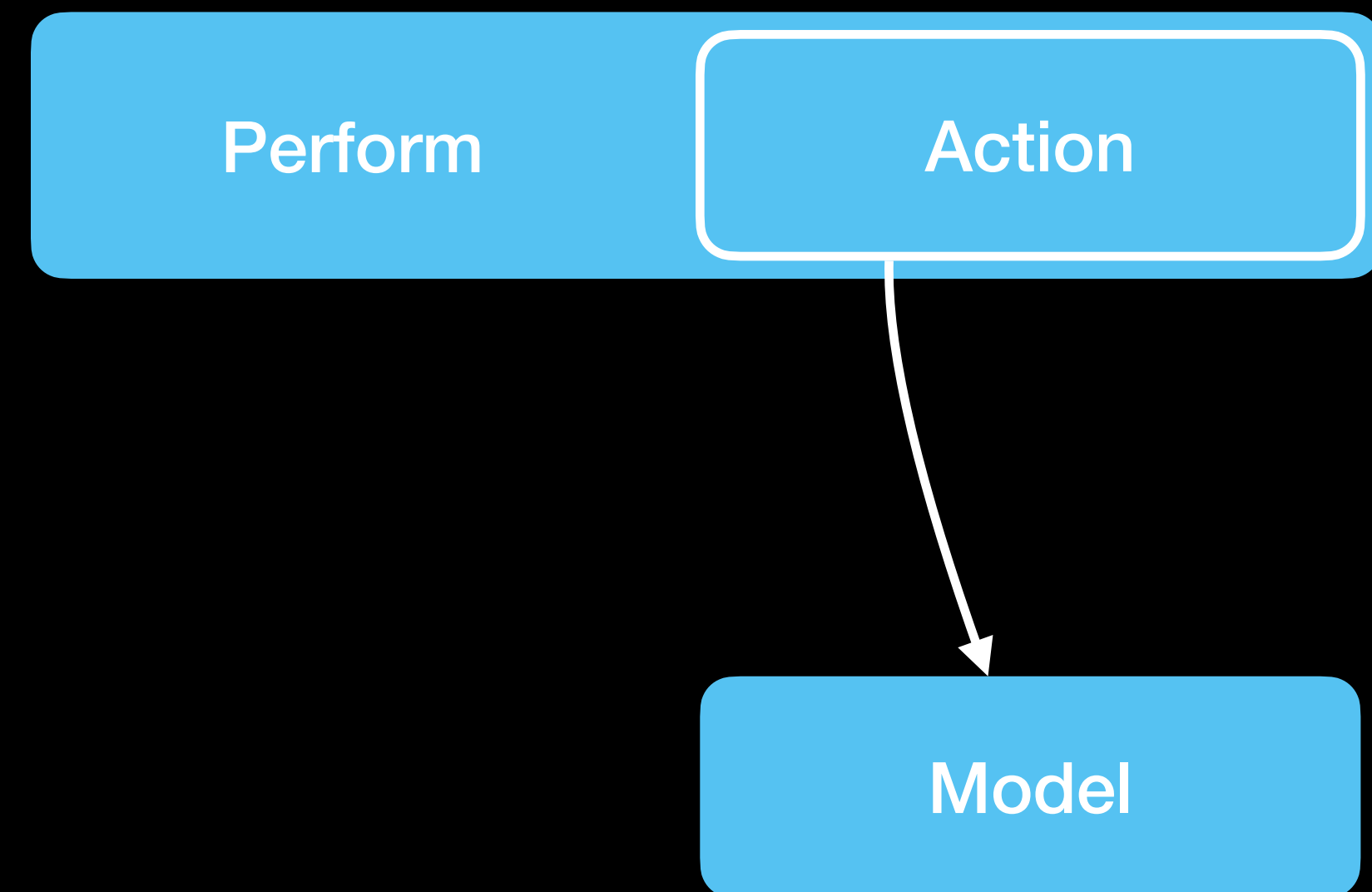
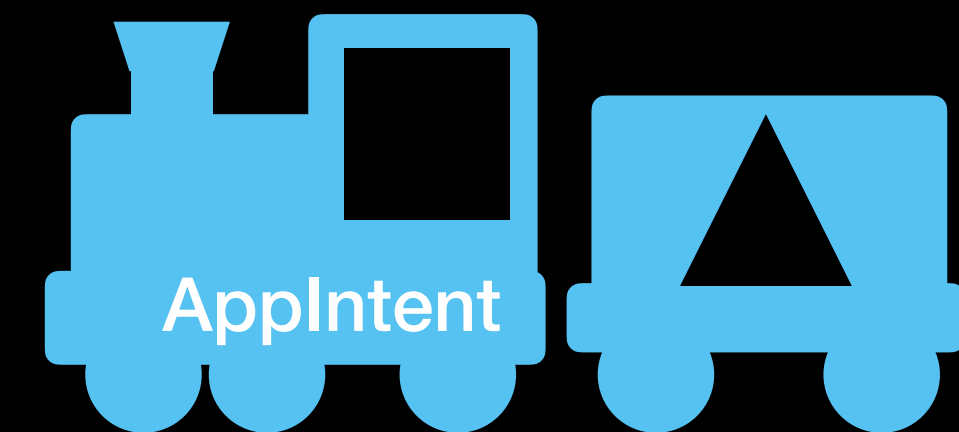
```

```

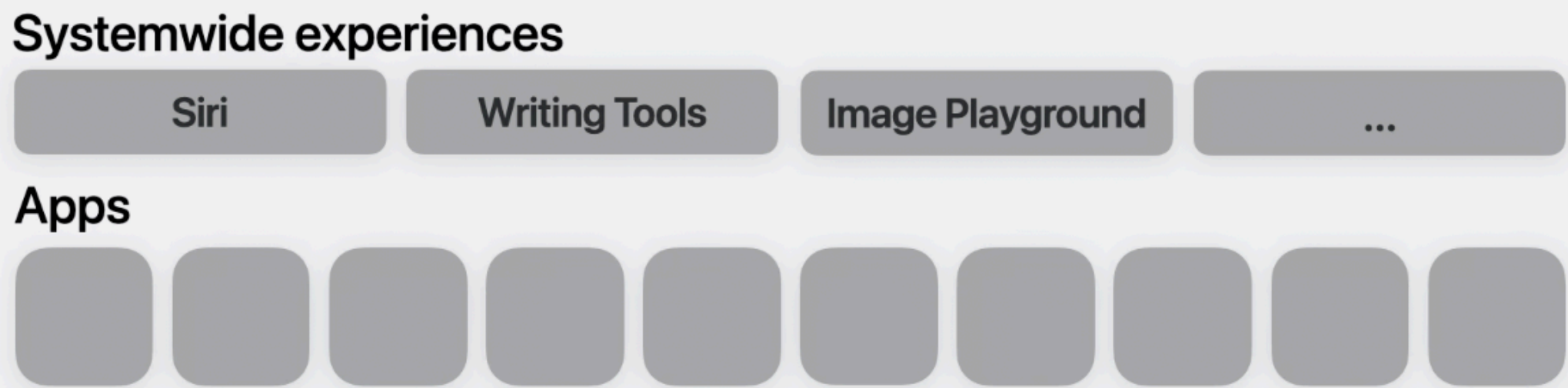
struct BabyIntent: AppIntent {
    @Dependency
    var model: Model

    func perform() async throws -> some IntentResult {
        ...
    }
}

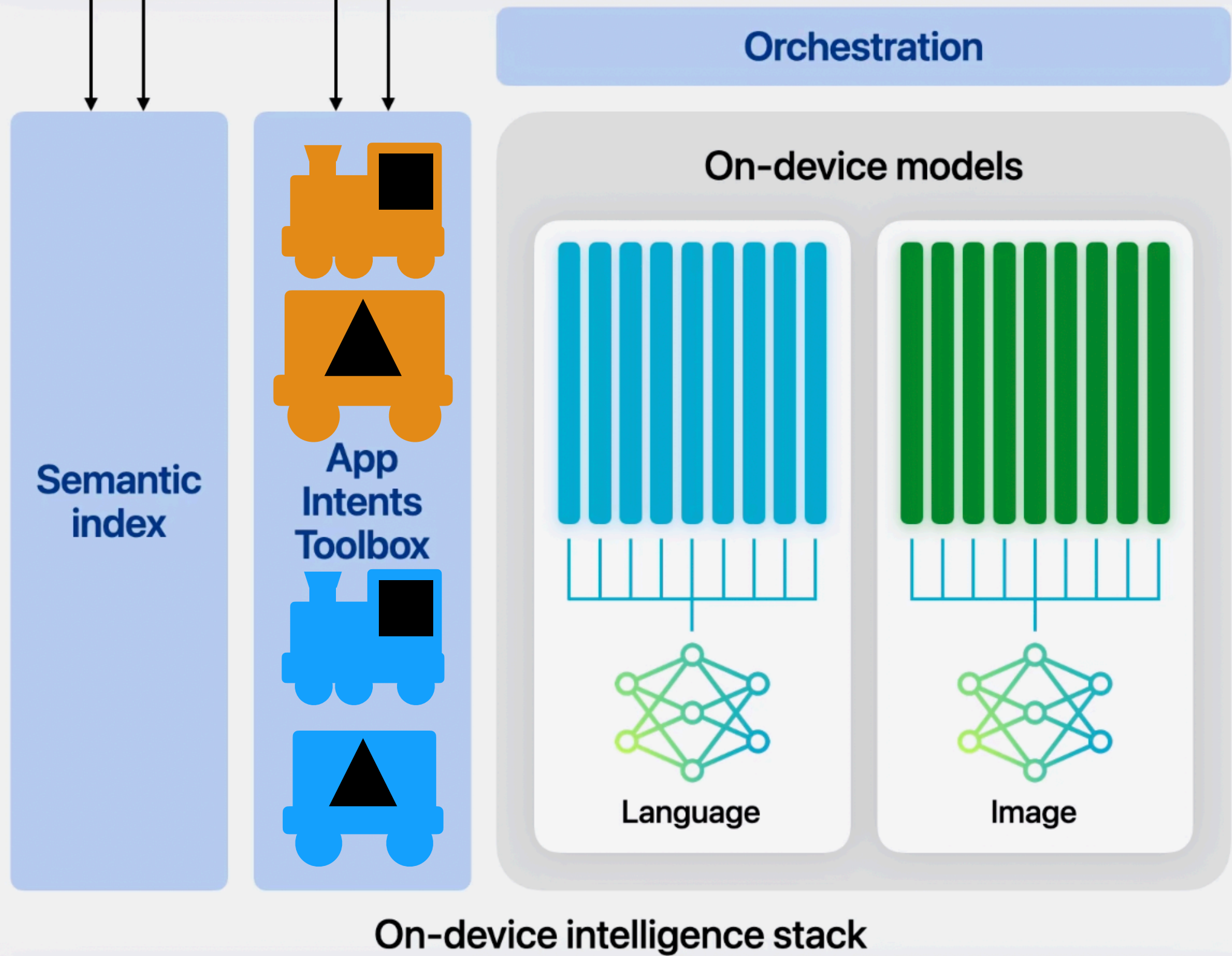
```



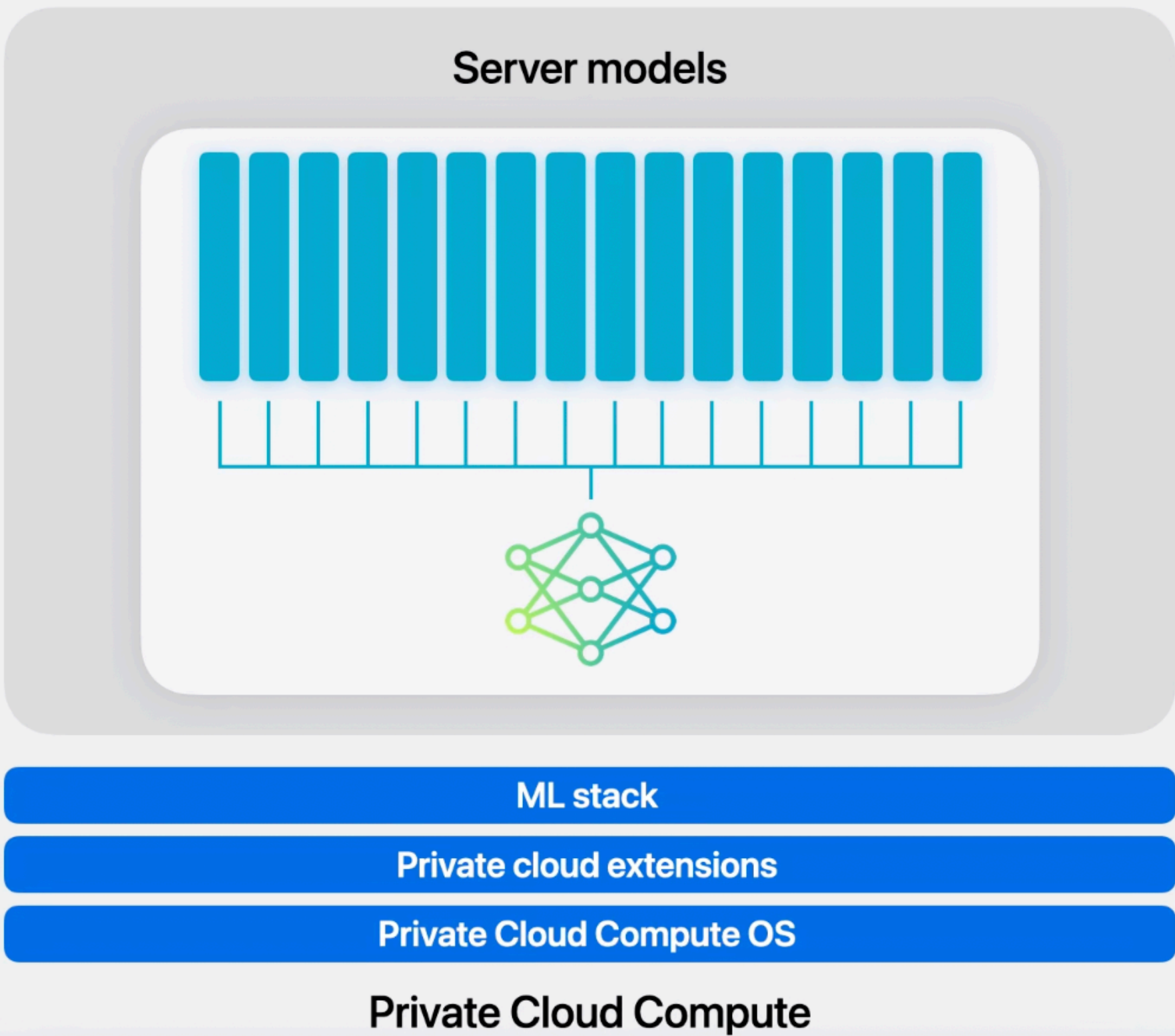
Apps and experiences



Personal Intelligence System



Apple silicon





WARNING



**Not recommended
for European developers
before 2025**





My First Writing Tools

MY FIRST
TOOLS



SEQUOIA

TOY STORE
ESTABLISHED 1877

SEQUOIA

TOY STORE
ESTABLISHED 1877

SEQUOIA

CHOIOO



RAG



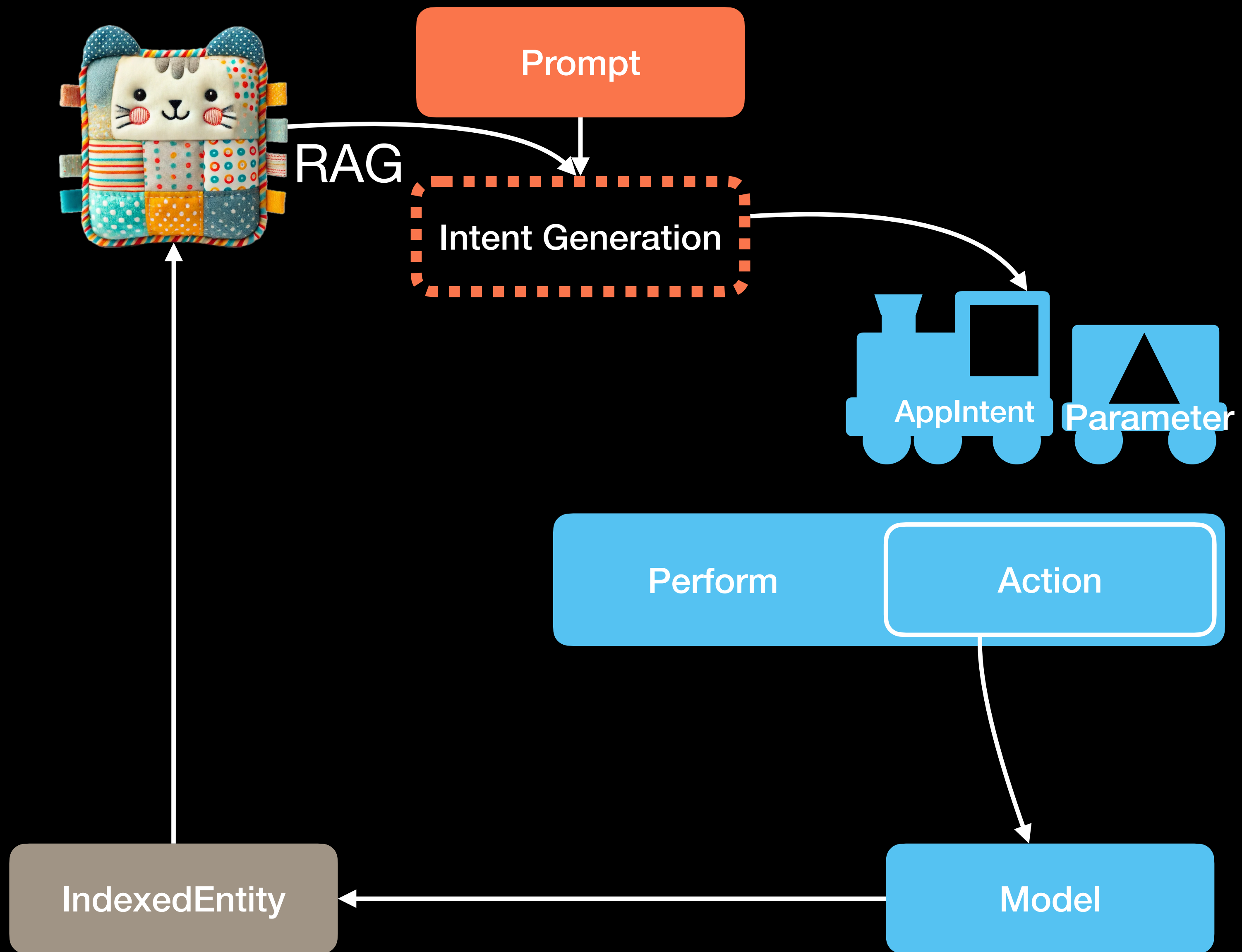
Find My First Entities



RAG

IndexedEntity

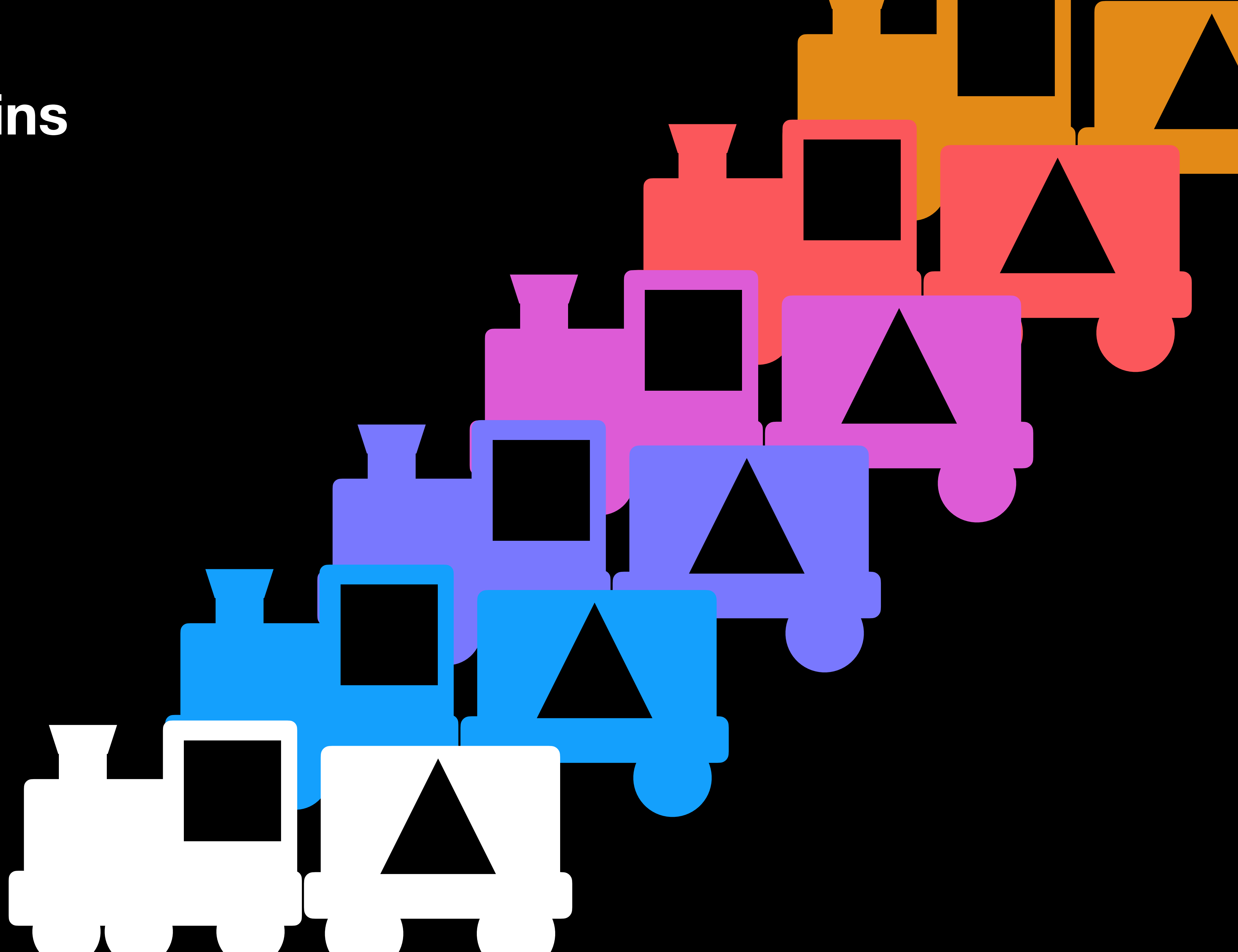
Model



```
try await CSSearchableIndex.default().indexAppEntities([entity])
```

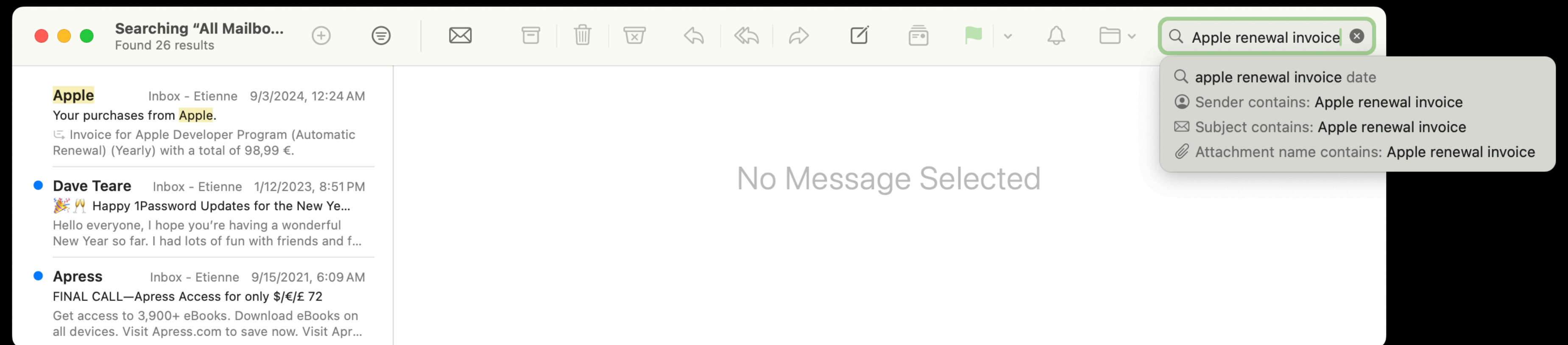

Apple Domains

In-app search



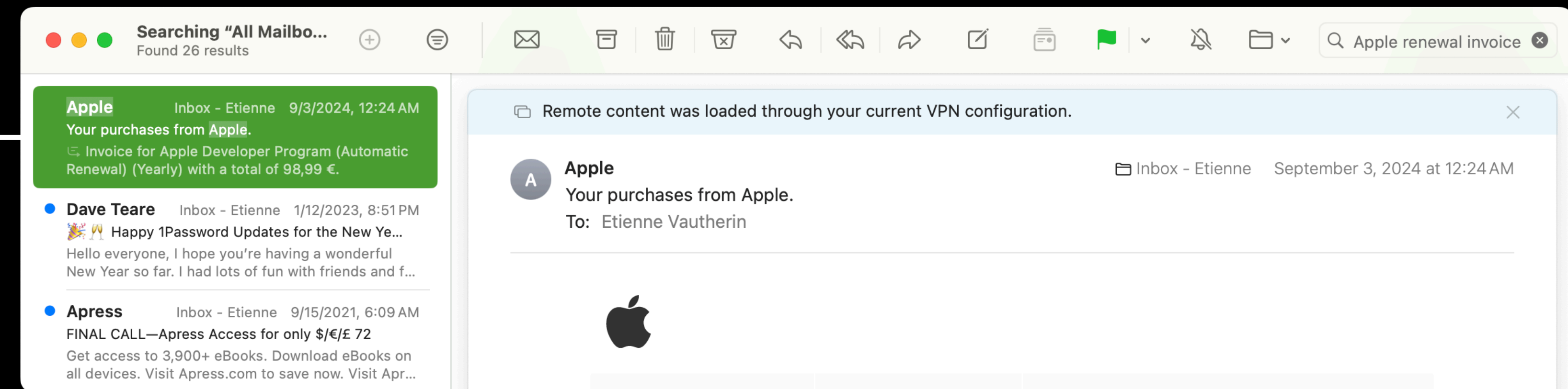


With in-app search, user adds some meaning to entities



IndexedEntity

User choice for
"Apple renewal
Invoice"





RAG

When is moms flight landing



Intent Generation

Answer

AppIntent

Parameter

Flight Tracker

Perform

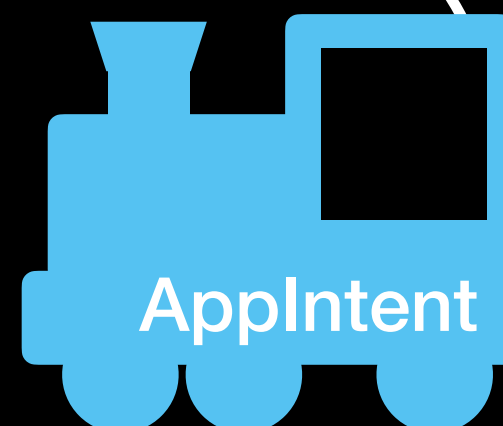
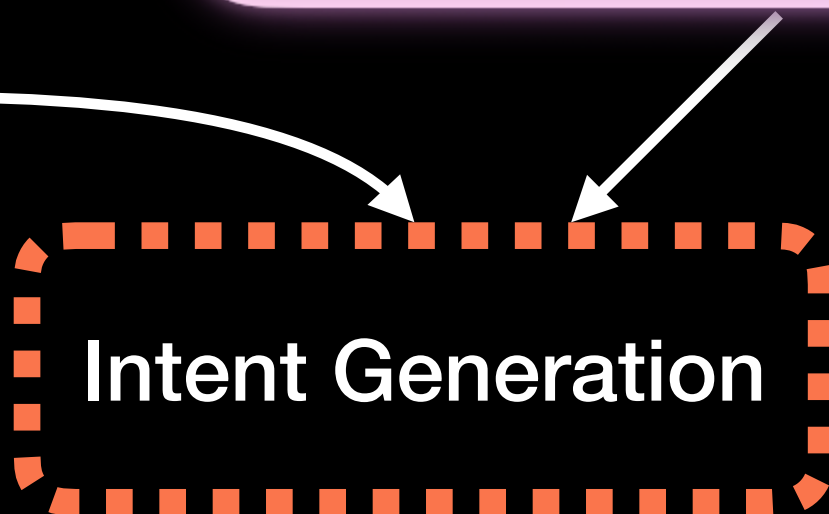
Description

Entity Result

TransferRepresentation

IndexedEntity

User choice for
"Moms flight »




```
// Set the lastUsedDate when the user interacts with the item
item.attributeSet.lastUsedDate = Date.now
item.isUpdate = true
```

```
// Interactions with items and suggestions from a query
query.userEngaged(engagedItem, visibleItems: visibleItems,
    interaction: CSUserQuery.UserInteractionKind.select)
query.userEngaged(suggestion, visibleSuggestions: visibleSuggestions,
    interaction: CSUserQuery.UserInteractionKind.select)
```




```
struct BabyEntityQuery: EnumerableEntityQuery {  
    @Dependency  
    private var model: Model  
  
    func entities(for identifiers: [BabyEntity.ID]) async throws -> [BabyEntity] {  
        ...  
    }  
  
    func suggestedEntities() async throws -> [BabyEntity] {  
        ...  
    }  
  
    func allEntities() async throws -> [BabyEntity] {  
        ...  
    }  
}
```



```

struct MyPhotoQuery: EntityPropertyQuery {
    static var properties = QueryProperties {
        Property(\.$takenAt) {
            LessThanMapping { URLQueryItem(name: "takenBefore", value: $0) }
            GreaterThanMapping { URLQueryItem(name: "takenAfter": value: $0) }
        }
        Property(\.$tags) {
            ContainsComparator { URLQueryItem(name: "tagsContains", value: $0) }
        }
    }
}

func entities(
    matching comparators: [URLQueryItem],
    mode: ComparatorMode,
    sortedBy: [Sort<MyPhoto>],
    limit: Int?
) async throws -> [MyPhoto] {
    let components = URLComponents()
    components.queryItems = comparators
    let url = components.url(relativeTo: "https://myphotosbackend.com/photos")

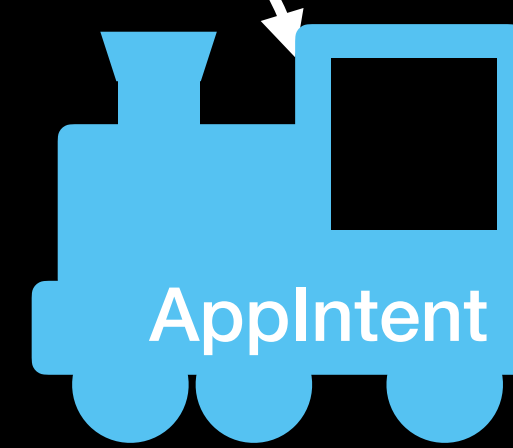
    return try await PhotosBackend.fetch(url: url)
}

```



🔊 Add a Pokemon weighting more than 10 to the last slide 🔊

Intent Generation



Perform

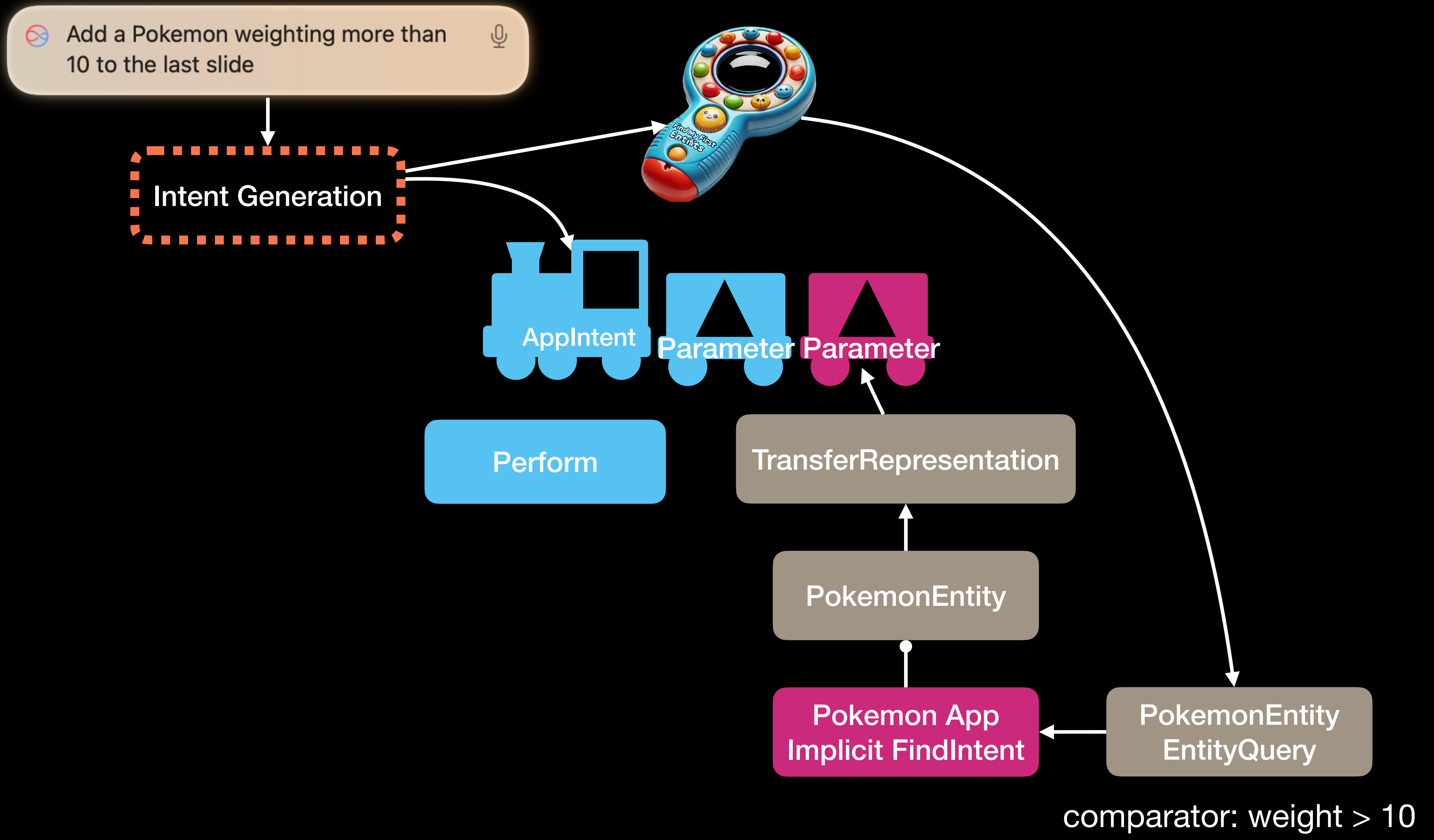
TransferRepresentation

PokemonEntity

Pokemon App
Implicit FindIntent

PokemonEntity
EntityQuery

comparator: weight > 10



This is just the beginning

A talk just for you

```
private struct Session {  
    let location = "online"  
    let duration: Duration = .minutes(30)  
    let cost = 0.0  
  
    // Book here  
}
```

