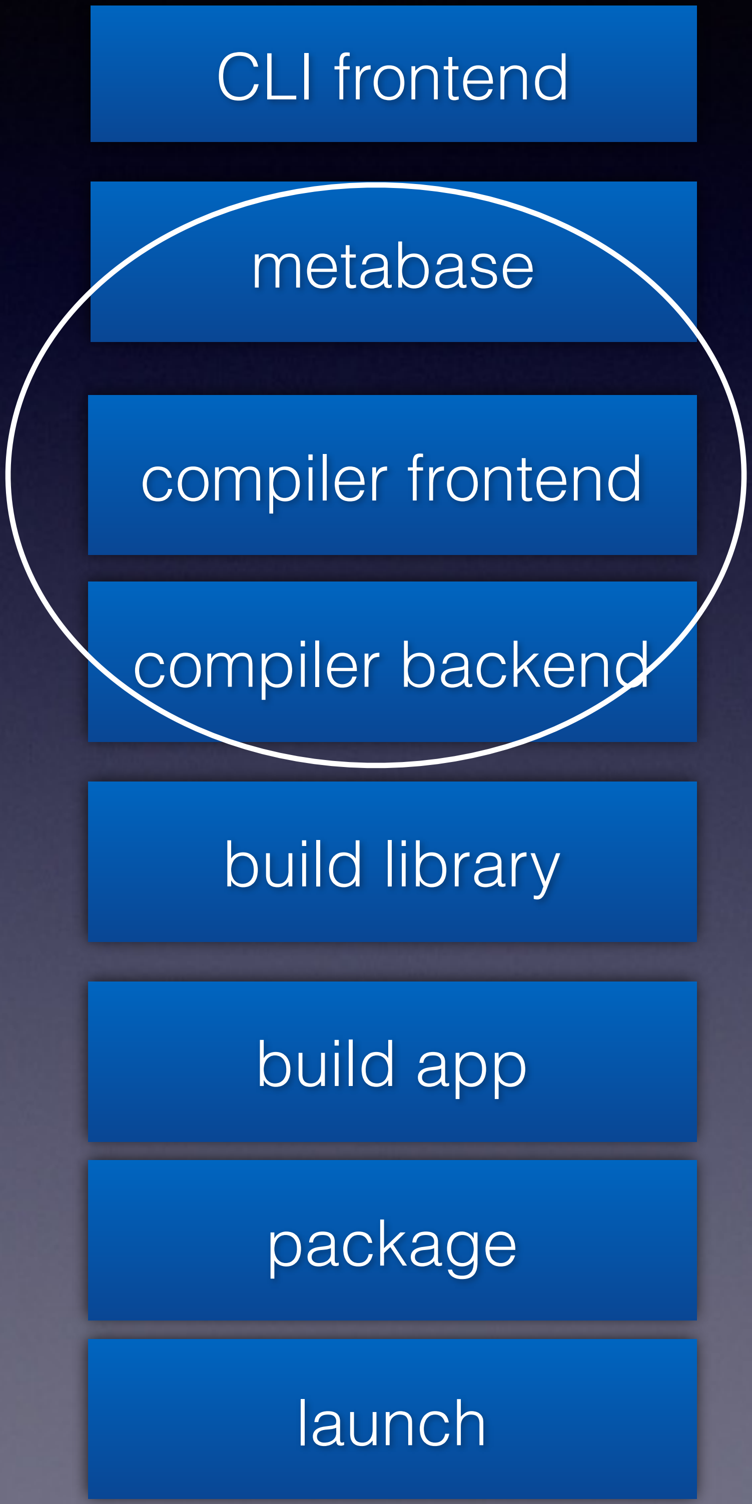
Hyperloop re-structuring

for 5th gen

Hyperloop components



CLI frontend

metabase

compiler frontend

compiler backend

build library

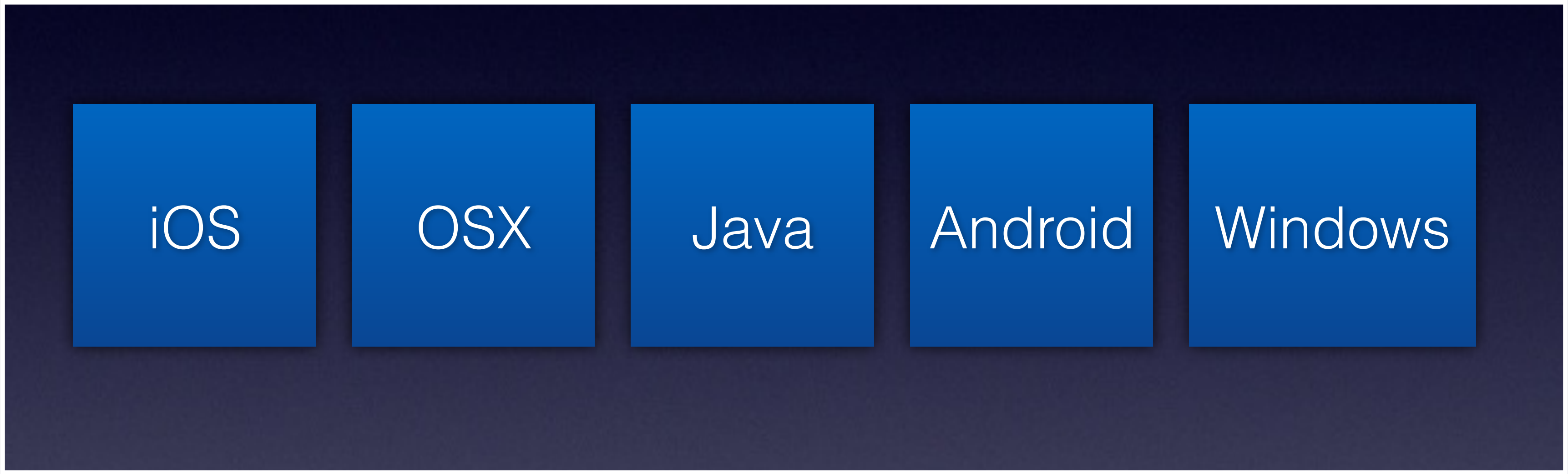
build app

package

launch

Decoupling

Define common metabase structure



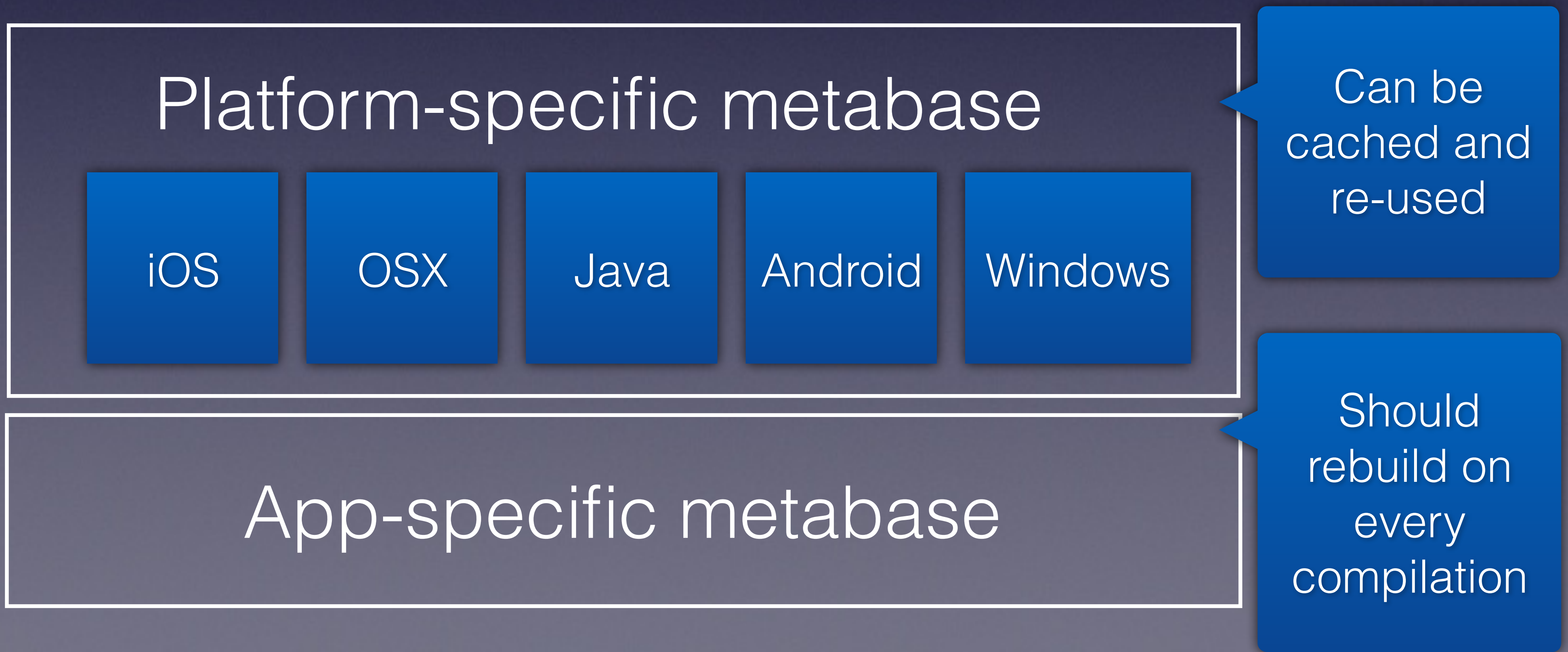
+ platform specific extension

This will really helps to reduce platform-specific code in compiler

app-specific metabase

- developer should be able to import their class libraries/header files and metabase should be able to build them separately

compiler arg —include=‘header.h’

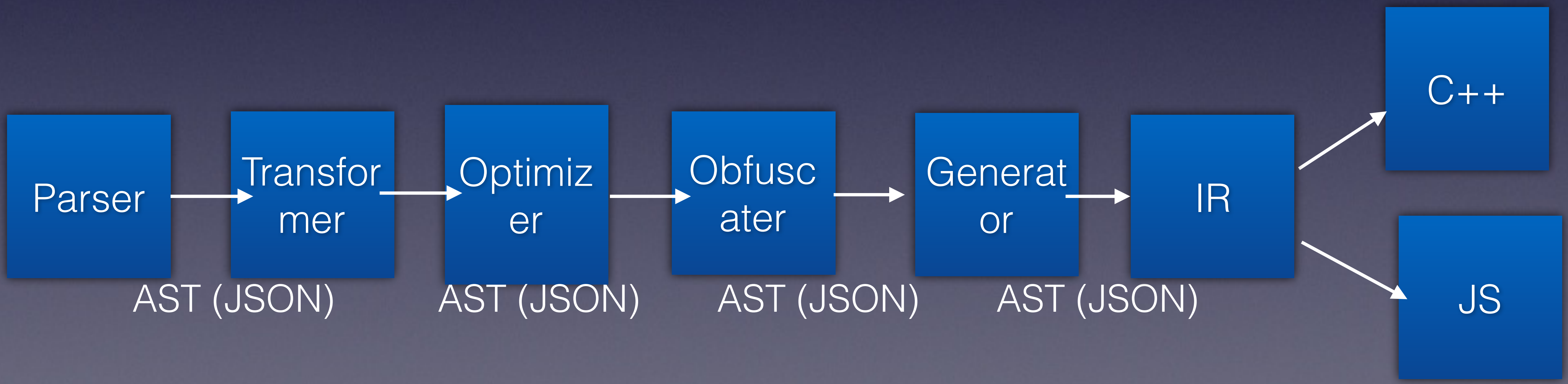


Can be cached and re-used

Should rebuild on every compilation

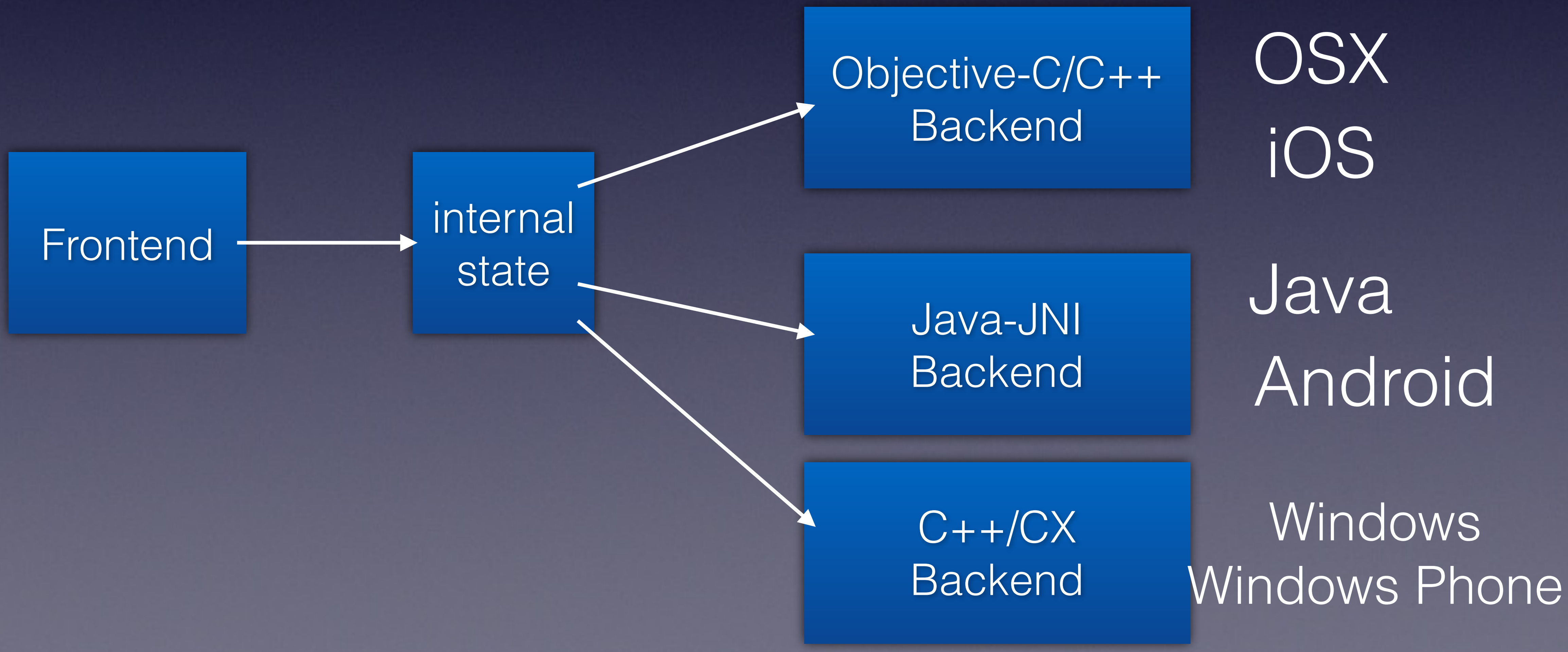
Well-defined, common JavaScript AST in JSON

- Mozilla JavaScript AST
- http://aosd.net/2013/escodegen.html



Frontend should focus on generating internal representation

- Front end compiler should not generate final source code or files like custom class etc.



Frontend

internal state

Objective-C/C++ Backend

OSX
iOS

Java-JNI Backend

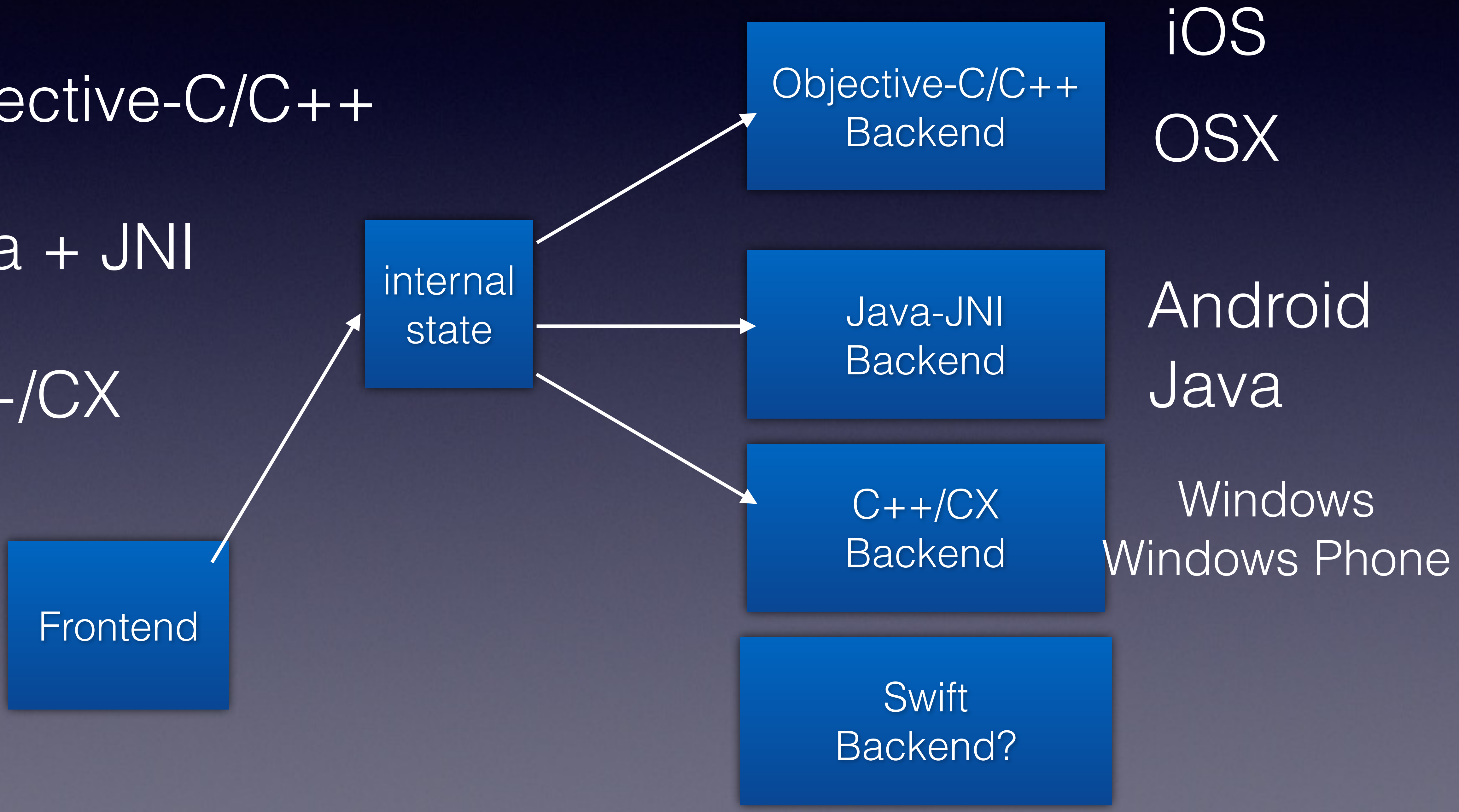
Java
Android

C++/CX Backend

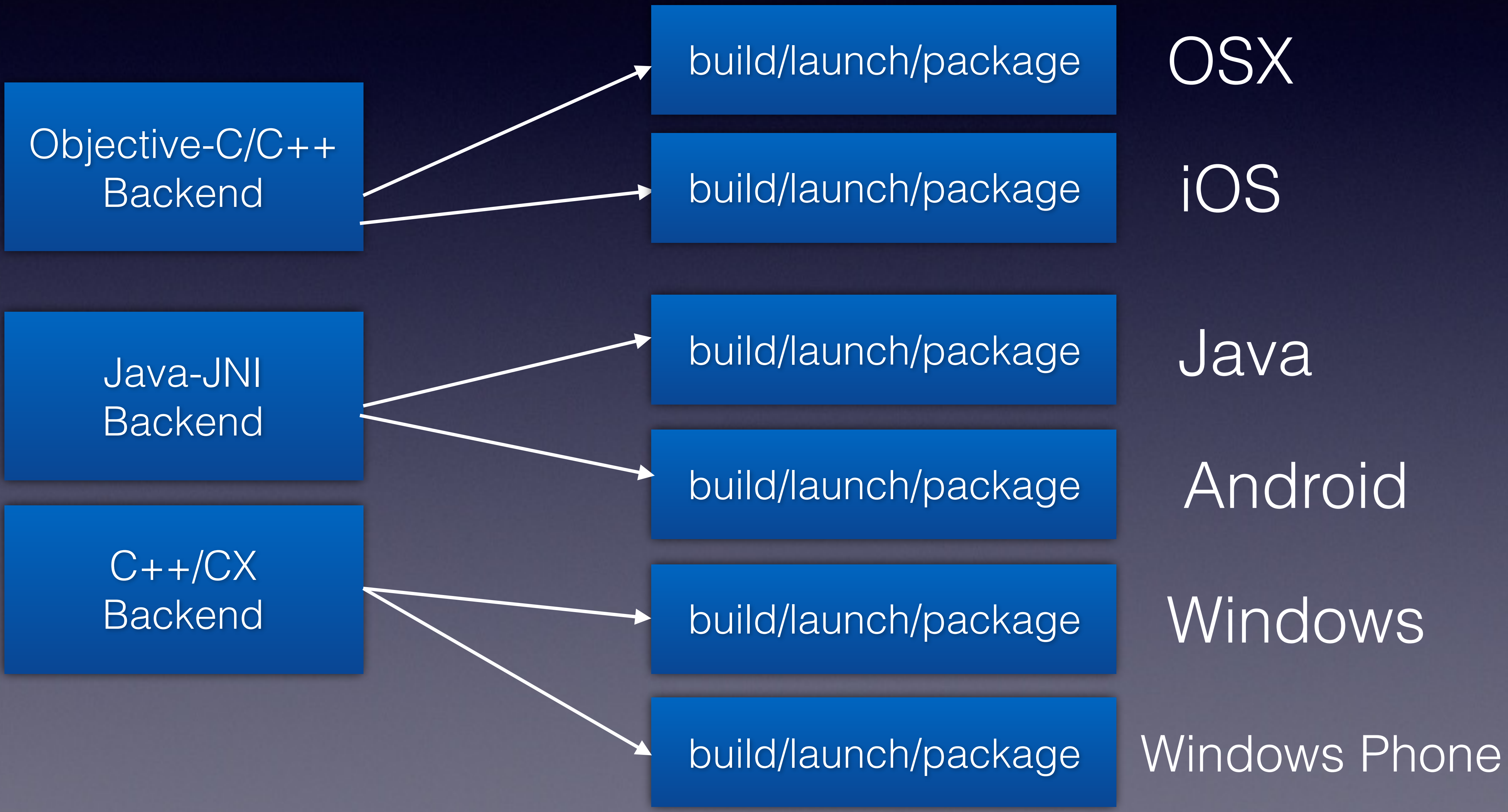
Windows
Windows Phone

Backend compiler should target language, not platform

- Objective-C/C++
- Java + JNI
- C++/CX



Builder/packager/launcher should target platform



Objective-C/C++ Backend

Java-JNI Backend

C++/CX Backend

build/launch/package OSX

build/launch/package iOS

build/launch/package Java

build/launch/package Android

build/launch/package Windows

build/launch/package Windows Phone

Extension

- Meant to eliminate Hyperloop.* functions
- Accepts JavaScript AST and extends global object

“use hyperloop.cast”

var a = Hyperloop.method(foo,’bar(**int**)’).call(i);

↓

var a = foo.bar(**int**(i));

Extension

- Meant to eliminate Hyperloop.* functions

“use hyperloop.cast”

var a = Hyperloop.method(foo,’bar(**com.myClass**)’).call(obj);

↓

var a = foo.bar(**cast**(‘com.myClass’,obj));