DESIGN + BUILD CASE STUDY

# WOLF ASCENT

## From cockpit to mobile play.

A helicopter action game that brings combat, rescue and convoy escort into one cockpit-led experience, with a buildout spanning desktop and mobile.




Real HELIWOLF gameplay in Flight School on Android build 33.

**910**

**first-party C# files**

One app-owned source snapshot

**158.9K**

**physical source lines**

Includes comments and blank lines

**10**

**campaign sorties**

Combat, rescue and convoy escort

# Clarity from briefing to cockpit.

Make a busy aircraft game readable on a tablet: show players where to go, what matters now and how to act without leaving the flight view.




Desert Run briefing: a selected mission, a route preview, clear objectives and touch-control reminders.

01

## Clear briefings

The mission directory puts the task, route and start action together before takeoff.

02

## A focused flight view

Heading, altitude, weapons and target information stay close to the action.

03

## Touch controls that fit

Movement and altitude controls sit alongside the actions a pilot needs now.

# Code that supports the next mission.

Flight, mission logic, world loading and player progress each have a defined job. Dedicated authoring tools and test code support content creation alongside the gameplay itself.

**300**
**runtime C# files**
Gameplay and flight foundation

**394**
**Editor / build files**
Authoring and delivery tools

**216**
**test source files**
A dedicated verification layer

**PLAYER COMMANDS + FEEDBACK**
Touch / keyboard input, cockpit displays and mission menus

**MISSION FLOW**
Selection, flight, objectives, debrief and retry

**FLIGHT + COMBAT**
Aircraft handling, weapons, projectiles and countermeasures

**WORLD + SCENERY**
Scenery loading, forest rendering and terrain contacts

**PROGRESS + INTERFACE**
Mission conditions, player progress and clear feedback

**CONTENT TOOLS + AUTOMATED CHECKS**
Six assembly definitions organize runtime, tooling and tests.

Counted once across app-owned source code, excluding third-party dependencies and duplicate platform copies. Physical source lines include comments and blank lines.

# A lighter start for a large world.

Coordinating forest rendering, collision and data release preserved authored world content while reducing memory allocated as Night Courier first appears.

**95.4%**

**less memory allocated when Night Courier first appears**

BUILD 32

**2.48 GB**

BUILD 33

**113.6 MB**

Measured initial scene allocation on the same Samsung tablet, comparing builds 32 and 33. This is not total app memory.

### 1.31 million authored tree instances / 16 terrain data sets




Night Courier: two normal loads observed, with approximately 19 minutes and 6.3 km in the first flight.

# Delivery that reaches real devices.

A consistent game experience has to travel through platform packaging, large content sets and updates that respect the player's existing progress.

**611**
**content hashes verified**

**75**
**device package files**
Exact delivered-file checks

**7**
**native libraries checked**

The Android build was checked from cooked content through signed packages to tablet installation. Its in-place update retained the app identity, data directory and visible player progress.




Daylight cockpit on Android build 33. The same product vision extends across mobile, tablet and desktop build targets.

**BUILD TARGETS / ANDROID / iOS + iPadOS / macOS**