3D ANALOGICAL FLIGHT INSTRUMENTS

High Fidelity

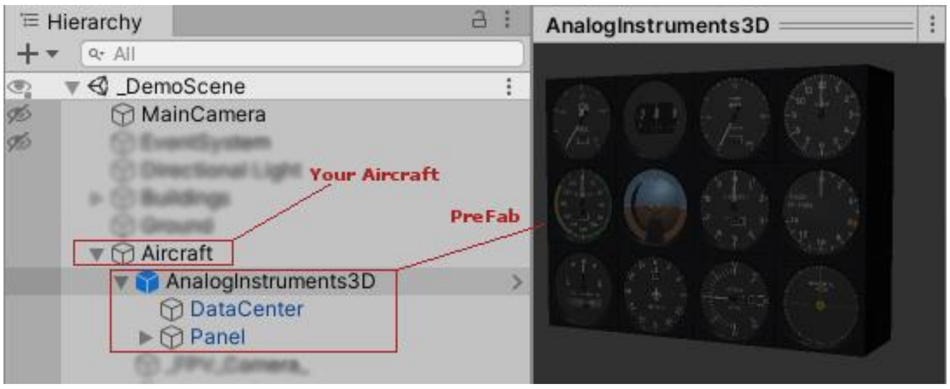
v 1.0

- **Contact: Maloke7-Games@yahoo.com.br**
- **Full Portfolio:** https://maloke.itch.io/
- **AssetStore:** https://assetstore.unity.com/publishers/26634

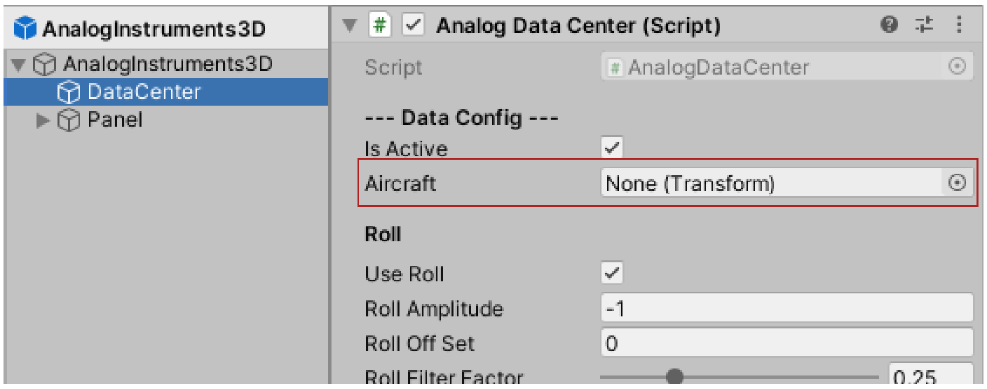
➢ Quick Instructions

To start using the instruments just follow these simple steps:

- Drop the "***AnalogInstruments3D***" PreFab inside your aircraft's Transform:



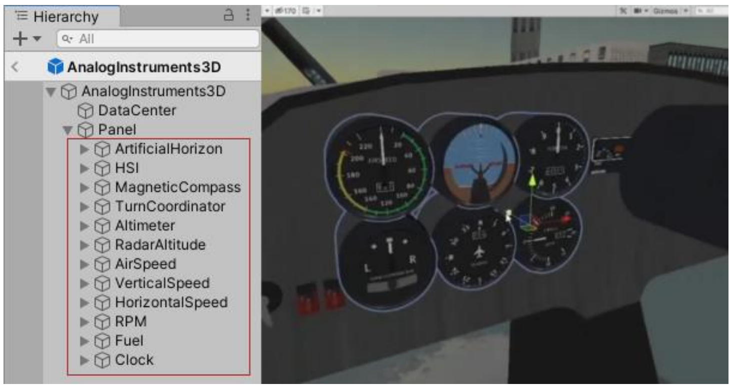
- Select the ***AnalogDataCenter*** script and link your aircraft's Transform:



Script: AnalogDataCenter
--- Data Config ---
Is Active: ✓
Aircraft: None (Transform)
Roll
Use Roll: ✓
Roll Amplitude: -1
Roll Off Set: 0
Roll Filter Factor: 0.25

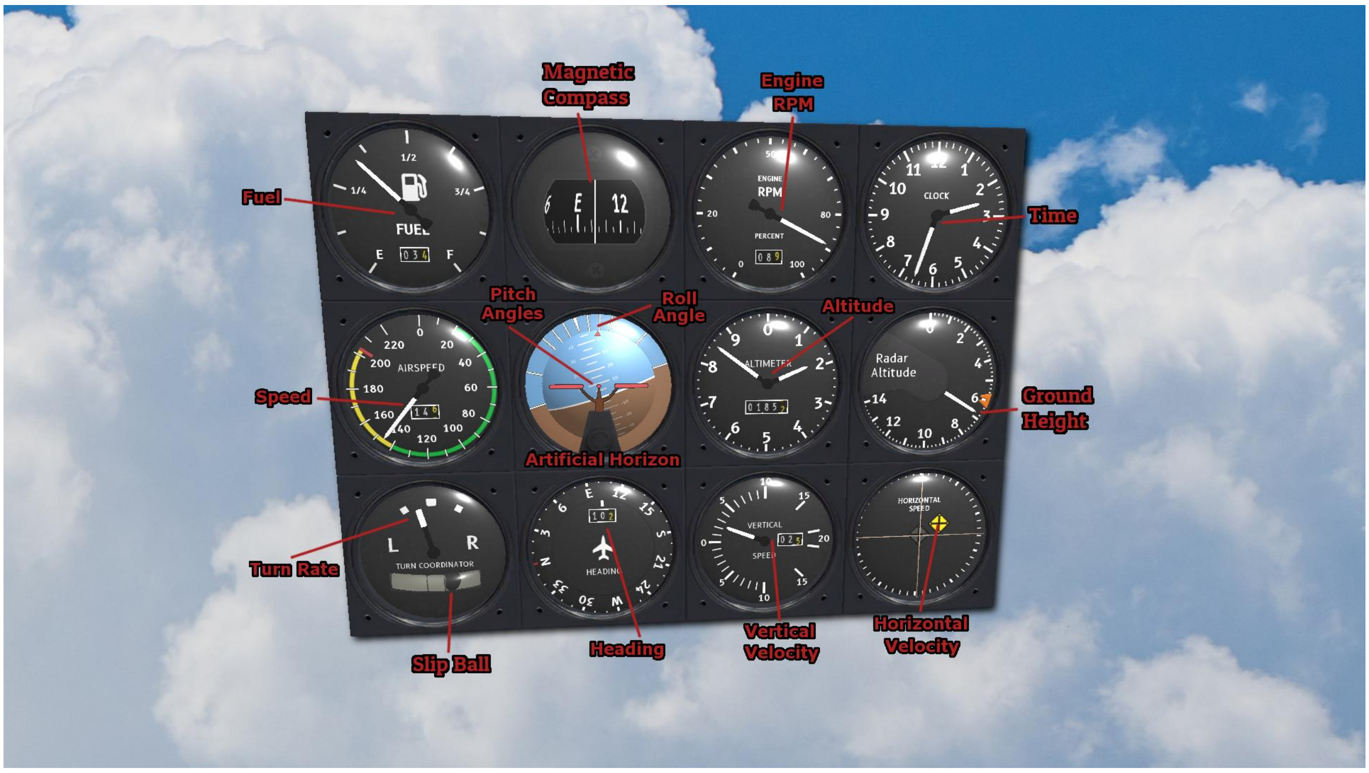
If you leave this field empty, the script will automatically track the movement of the **MainCamera instead. Ideally, for better and more realistic reading, you should link your aircraft's **center of mass** here. Another option is to link the **DataCenter**'s Transform itself and then move it to overlap the aircraft's center of mass position.*

- Inside the ***Panel***, you will find all the instrument's gameobjects. Now displace them through your panel and you are ready to fly:



****If you want to know more about this asset you can find extra information further on this document.***

➢ Instruments Symbology



At the moment we have a total of 12 different instruments (from top left to right down):

- Fuel Indicator (%)

- Magnetic Compass: Heading

- Engine RPM (%)

- Clock: Can display SystemTime or Custom Time

- AirSpeed

- Artificial Horizon: Pitch and Roll angles

- Altimeter: Indicates Altitude

- Radar Altitude: Indicates Ground Height

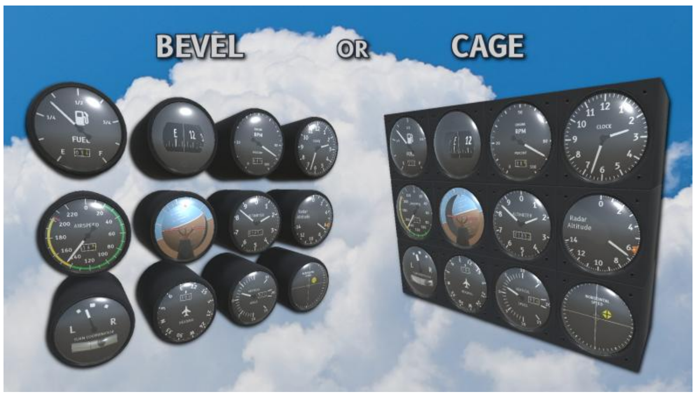
- Turn Coordinator: Indicates Turn-Rate (Needle) and Slip Angle (Ball)

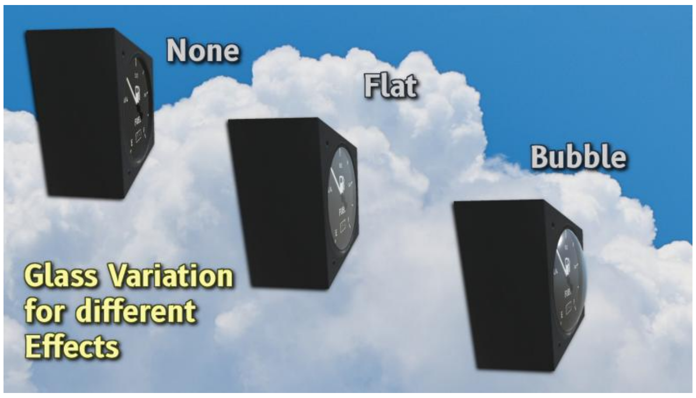
- HSI (Horizontal Situation Indicator): Show current Heading

- Vertical Speed: Indicates if the aircraft is going up or down

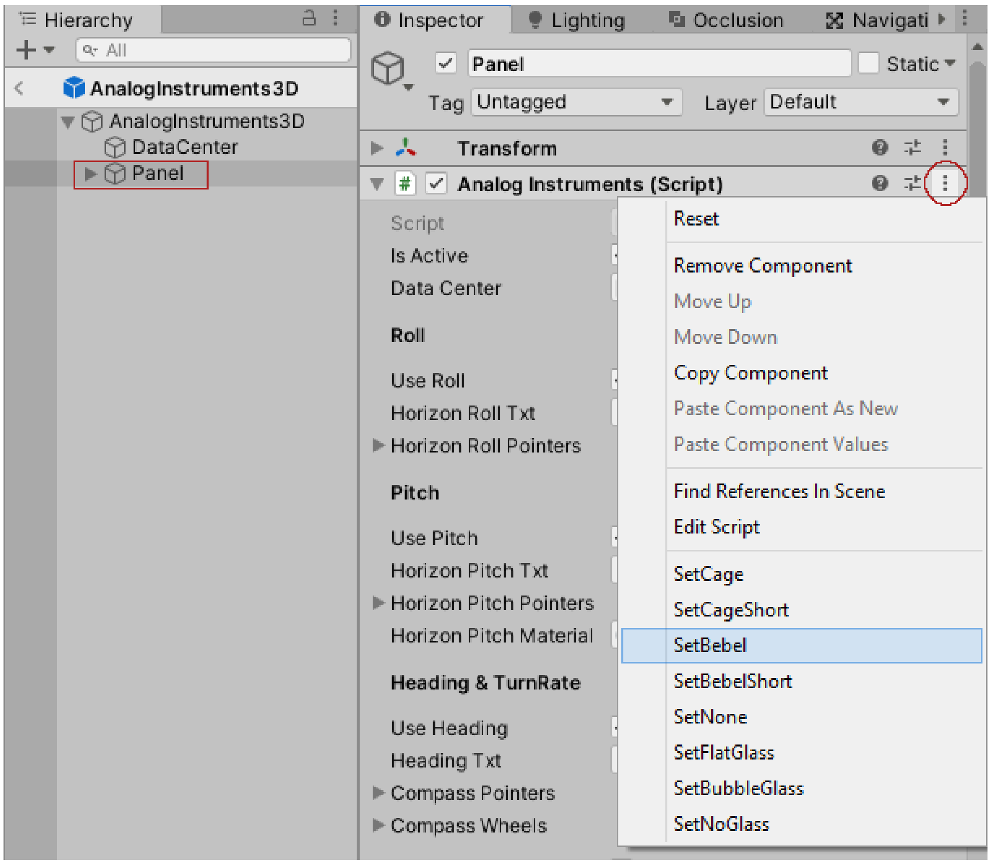
- Horizontal Speed: Indicates forward/backward and lateral speed (Useful for landing helicopters)

➢ Customization: Bevel/Cage and Glass





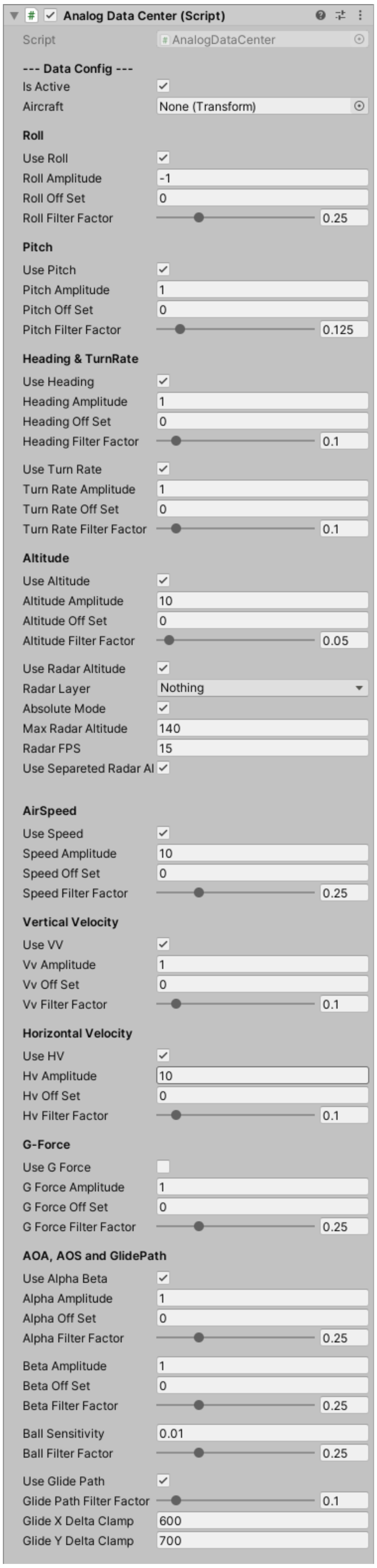
To change/customize between multiple Bevel/Cage styles and Glass variations, just select the ***Panel*** and click on the ***ContextMenu*** of the ***AnalogInstruments*** script:



Obs: The ***AnalogInstruments*** script is responsible for reading the flight variables value from the ***AnalogDataCenter*** script and passing it to each instrument's pointers, needles, and indicators. If you are using the Template PreFab there is no need to bother, everything is already linked for you.

➢ AnalogDataCenter Script

The ***AnalogDataCenter*** script is responsible for calculating each flight variables value from the provided aircraft's Transform. You can tweak, scale and offset these values. Below you will find a complete description of each element of this script:



Analog Data Center (Script)

Script	AnalogDataCenter
--- Data Config ---	
Is Active	✓
Aircraft	None (Transform)
Roll	
Use Roll	✓
Roll Amplitude	-1
Roll Off Set	0
Roll Filter Factor	0.25
Pitch	
Use Pitch	✓
Pitch Amplitude	1
Pitch Off Set	0
Pitch Filter Factor	0.125
Heading & TurnRate	
Use Heading	✓
Heading Amplitude	1
Heading Off Set	0
Heading Filter Factor	0.1
Use Turn Rate	✓
Turn Rate Amplitude	1
Turn Rate Off Set	0
Turn Rate Filter Factor	0.1
Altitude	
Use Altitude	✓
Altitude Amplitude	10
Altitude Off Set	0
Altitude Filter Factor	0.05
Use Radar Altitude	✓
Radar Layer	Nothing
Absolute Mode	✓
Max Radar Altitude	140
Radar FPS	15
Use Separeted Radar Al	✓
AirSpeed	
Use Speed	✓
Speed Amplitude	10
Speed Off Set	0
Speed Filter Factor	0.25
Vertical Velocity	
Use VV	✓
Vv Amplitude	1
Vv Off Set	0
Vv Filter Factor	0.1
Horizontal Velocity	
Use HV	✓
Hv Amplitude	10
Hv Off Set	0
Hv Filter Factor	0.1
G-Force	
Use G Force	
G Force Amplitude	1
G Force Off Set	0
G Force Filter Factor	0.25
AOA, AOS and GlidePath	
Use Alpha Beta	✓
Alpha Amplitude	1
Alpha Off Set	0
Alpha Filter Factor	0.25
Beta Amplitude	1
Beta Off Set	0
Beta Filter Factor	0.25
Ball Sensitivity	0.01
Ball Filter Factor	0.25
Use Glide Path	✓
Glide Path Filter Factor	0.1
Glide X Delta Clamp	600
Glide Y Delta Clamp	700

- "**isActive**" determines if the script should be active.

- "**Aircraft**"(***Transform***) is the reference to your flying gameObject which will be used to calculate all the values displayed on the instruments.

Each section below corresponds to a ***Flight Variable*** and the following pattern applies to all of them:

-The *bool* values "**useXXX**" determine if that variable will be calculated by the script.

- The "**xxxAmplitude**" is a value multiplied to that variable after calculations and before showing on instruments. It can work as a ***Scale*** or as a ***unit conversion factor***.

- The "**xxxOffSet**" is a value added to the variable after calculation. It can be used as ***unit conversion***, to fix ***objects orientation*** properly or, for instance, to adjust altitude zero.

- All the "**xxxFilterFactor**" values are used to smooth the value shown *(works as a **lowpass filter*)**. If set to **1** it will ***show direct value*** and will have no filtering at all. If set near **0** it ***will take more time to reach the actual value***.

- The "**UseRadarAltitude**" bool will enable the script calculation for ground height.

- The layer "**RadarLayer**" can be used to ignore or include some objects like trees from the raycasting for detecting the ground.

- If "**AbsoluteMode**" is enabled the displayed height will be calculated independent of the aircraft attitude, otherwise, it will behave realistically detecting obstacles straight down relative to the aircraft attitude.

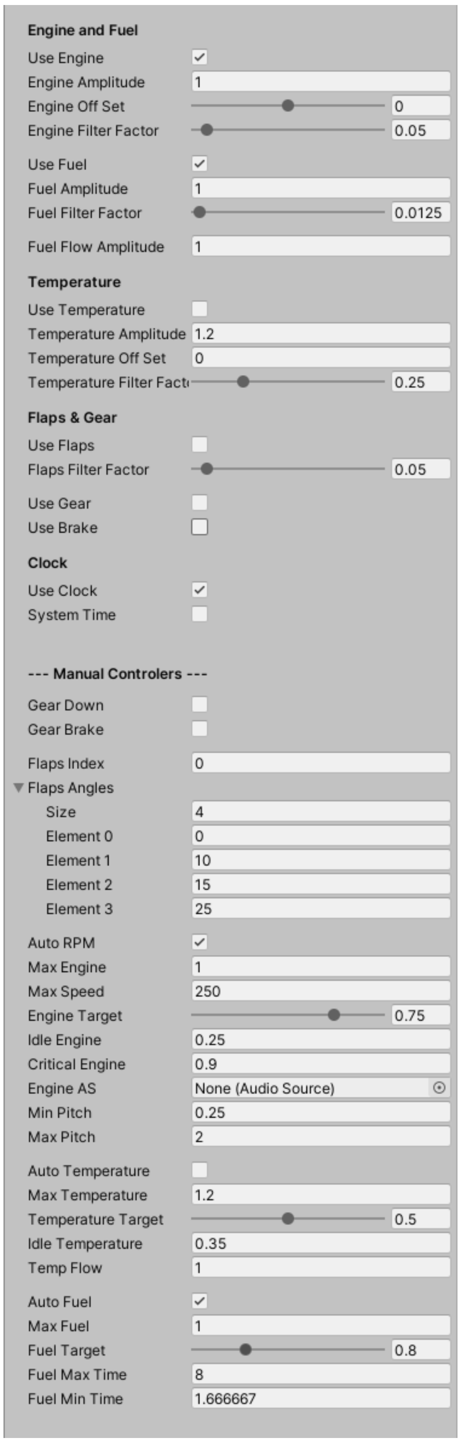
- "**MaxRadarAltitude**" indicates the maximum range the radar can detect the ground. If no obstacles are found within this range, the instrument's needle will display the máximum value.

- The "**UseSeparatedRadarAltitude**" bool will enable the extra instrument dedicated to showing the ground height. If this is disabled, then the default altimeter will display the ground height instead of the altitude.

- The **AOA** (***Angle of Attack***), **AOS** (***Angle of Slip***), and the **GlidePath** (*also called* ***Velocity Vector*** *or* ***flight path vector FPV***) are specific terms on aviation and engineering that are a bit complex to explain here, so here are some Wiki links with a more detailed explanation about them:

- https://en.wikipedia.org/wiki/Head-up_display#Displayed_data
- https://en.wikipedia.org/wiki/Angle_of_attack
- https://en.wikipedia.org/wiki/Slip_(aerodynamics)

- The "**Glide XY Delta Clamp**" determines how much the **Glide Path** indicator will be clamped from the center position. If you do not wish to clamp it inside a HUD/Gauge, then you can just set these to some large value.



Engine and Fuel	
Use Engine	✓
Engine Amplitude	1
Engine Off Set	0
Engine Filter Factor	0.05
Use Fuel	✓
Fuel Amplitude	1
Fuel Filter Factor	0.0125
Fuel Flow Amplitude	1
Temperature	
Use Temperature	☐
Temperature Amplitude	1.2
Temperature Off Set	0
Temperature Filter Fact	0.25
Flaps & Gear	
Use Flaps	☐
Flaps Filter Factor	0.05
Use Gear	☐
Use Brake	☐
Clock	
Use Clock	✓
System Time	☐
--- Manual Controlers ---	
Gear Down	☐
Gear Brake	☐
Flaps Index	0
Flaps Angles	
Size	4
Element 0	0
Element 1	10
Element 2	15
Element 3	25
Auto RPM	✓
Max Engine	1
Max Speed	250
Engine Target	0.75
Idle Engine	0.25
Critical Engine	0.9
Engine AS	None (Audio Source)
Min Pitch	0.25
Max Pitch	2
Auto Temperature	☐
Max Temperature	1.2
Temperature Target	0.5
Idle Temperature	0.35
Temp Flow	1
Auto Fuel	✓
Max Fuel	1
Fuel Target	0.8
Fuel Max Time	8
Fuel Min Time	1.666667

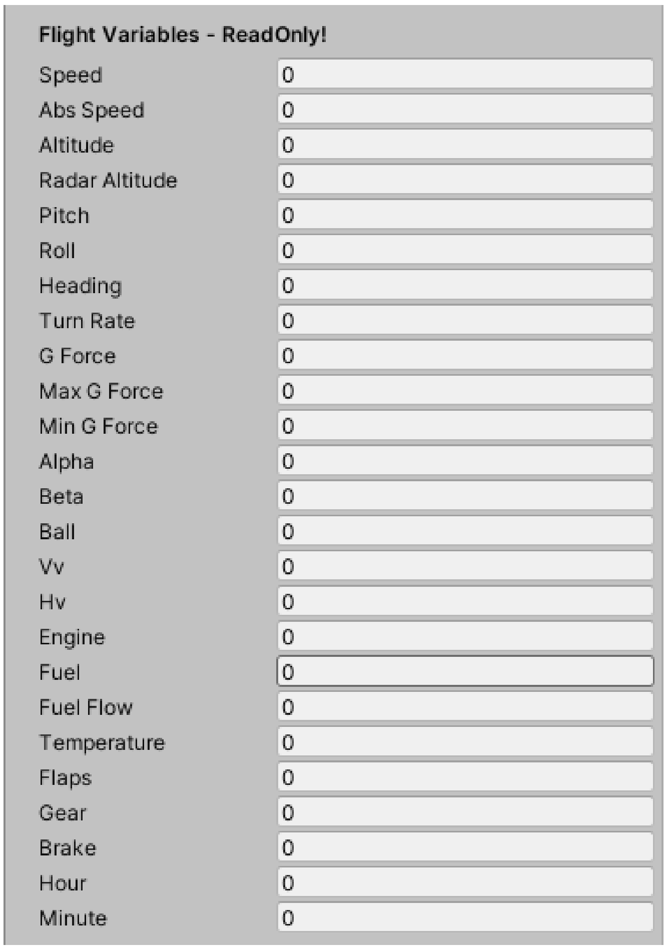
- The **Engine** and **Fuel** variables are **a percentage** from 0 to 1 (empty/full).

- On the **Manual Controllers** section you can manage the current **Engine RPM** and **Fuel quantity** using the Target sliders (from 0 to 1 for empty/full).

- If **AutoRPM** is off, then you can manually control the Engine RPM. If **AutoRPM** is on and **MaxSpeed** is set to non-*zero*, then the RPM will synchronize automatically to correspond 100% RPM to that airspeed and interpolates it to zero when stopped. The same applies to Temperature.

- If the **EngineAudioSource** is not null, then the source **audio pitch** will automatically follow from **Min** to **MaxPitch** value and thus simulating the engine RPM sound accordingly.

- The **IdleFuelFlow** and **MaxFuelFlow** determine how much ***% of the fuel is consumed for one minute***. The **IdleFuelFlow** determines *the minimum consumption*, while *Max* determines consumption ***when Engine RPM is at 100%*** and interpolates it in-between.



Flight Variables - ReadOnly!

Speed	0
Abs Speed	0
Altitude	0
Radar Altitude	0
Pitch	0
Roll	0
Heading	0
Turn Rate	0
G Force	0
Max G Force	0
Min G Force	0
Alpha	0
Beta	0
Ball	0
Vv	0
Hv	0
Engine	0
Fuel	0
Fuel Flow	0
Temperature	0
Flaps	0
Gear	0
Brake	0
Hour	0
Minute	0

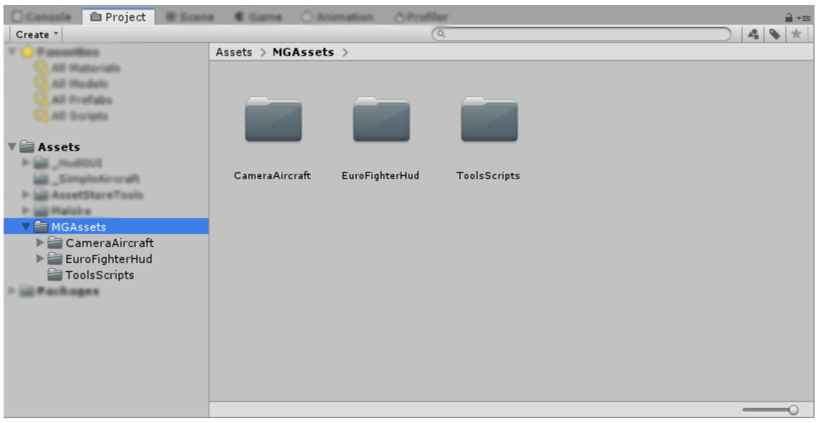
- On the **Flight Variables** section you can see all current values for all the variables in real-time inside the Editor.

This is just for debug or tweaking and are **ReadOnly**!

Note that the DataCenter is capable of calculating more data than currently available instruments, this gives room for future updates.

➢ Asset's Folders Organization:

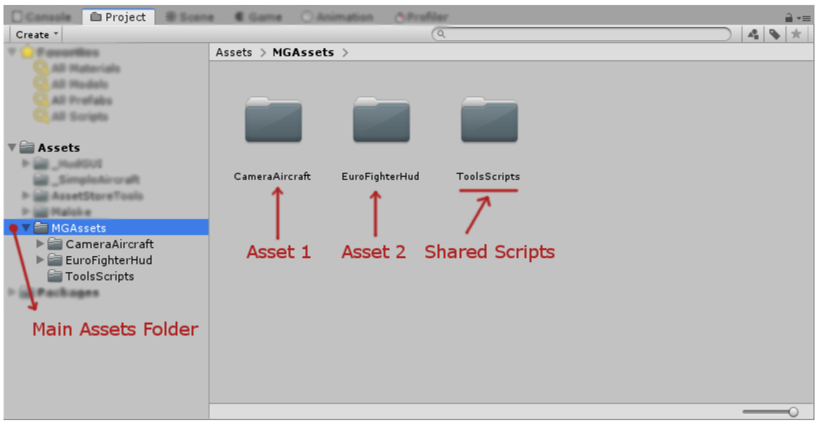
All assets and packages from MalokeGamesAssets will be downloaded/unpacked to a folder called **"MGAssets"** inside the Unity's **"Assets"** root:



Inside **"MGAssets"** you will find a separate folder for each asset package and all their specific resources (like data, scripts, textures, sprites, prefabs, demo scenes and so on...) will be found inside and organized in their respective subfolders.

Notice that some assets may use "under the hood" some general scripts and shared funcionalities, so for this reason (and to avoid duplicity or accidental deletion) you will find all this shared tools inside a folder called **"ToolsScripts"**.

Feel free to explore and use them on your projects too, they are simple but handy!


Project
Create
Assets > MGAssets >
Assets
MGAssets
CameraAircraft
EuroFighterHud
ToolsScripts
Asset 1
Asset 2
Shared Scripts
Main Assets Folder