Chapter 0: What?

Prerequisites & Disclaimers

I expect you to know how to use the Unity Editor and to know a little bit about vector math and geometry in general.
I wont go into too much detail for things that can be found in the Unity manual, but I will link to them when appropiate.

What is a Transform?

A transform is a component that is attached to a GameObject. In fact transforms are the only type of components that are always found on a GameObject, as every GameObject must have a transform and automatically comes with one when created.

This may make you wonder why transforms arent built into the GameObject itself. The reason is that there is two types of Transforms. The standard "Transform" and the "RectTransform" which is meant for use in UI.

In this guide I will exclusively talk about the standard "Transform", however "RectTransforms" have unique properties and abilities.

How do you do maths with a Transform?

For doing maths with transforms there are three primarily interesting things stored in a transform that we can manipulate: Position, Rotation and Scale.

The position is basically just a 3-dimensional vector we can manipulate. Therefore we can use it to do vector maths!
In the inspector, the position is always displayed in "local-space", so be aware of that when looking at your transforms in the inspector.

Rotations are less interesting for doing vector maths specifically, but generally the same principles used for the vector manipulations can be applied to rotations.

Scale seems to be interesting at first as it can be used to scale i.e. multiply the position of a child. However there is a much better way to achieve the same thing.

Position Constraints

Position Constraints are Unity built-in Components that will copy the world-space position of a single target, or blend between multiple positions when given multiple targets. They are a primary tool to manipulate Transforms, hence why I will briefly explain them here. Constraints generally have the following properties:

Is Active - This will simply activate/deactivate the constraint.

Weight - This is the overall weight of the constraint. This is a normalized value between 0 and 1 and will blend linearly between the "Position at Rest" and the final result of the constraint position.

Constraint Settings:

Lock - When locked, neither the "Position at Rest" nor the "Position Offset" will change. You generally want this to be locked, and Unity will automatically lock it in playmode.

Position at Rest - This is the position where the object rests aka when its not effected by the contraint. Generally this should be at (0,0,0)

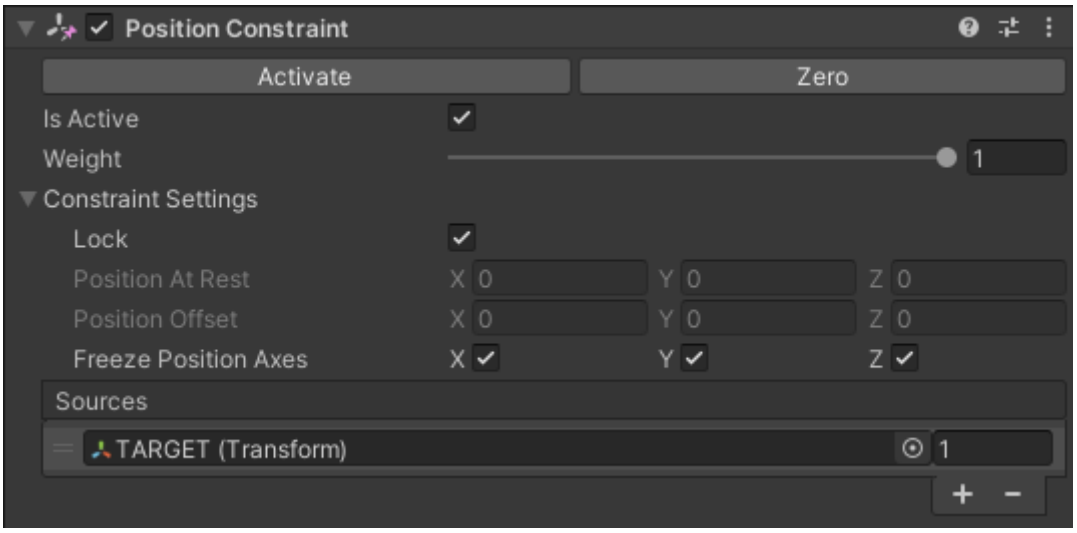
Position Offset - Is generally calculated from the "Position at Rest" and the constraint position. You generally dont need to adjust this. Generally this should also be at (0,0,0)

Freeze Position Axes - Terribly named, simply determines which axes are effected by the constraint.

Sources - This is the list that will contain all of the target transforms and their associated weight. Important to note here is that the weights will be normalized, when there is more than one target.

Be aware that Constraints are active even when the Editor is in "Edit Mode". More importantly, changes made to a transform by a constraint, wont be recorded by the Undo system. This means that you can permanently break things, when messing with constraints rather easily.

This is how a standard PositionConstraint configuration looks like:



For more information on (Position) Constraints look in the Unity Manual: https://docs.unity3d.com/Manual/class-PositionConstraint.html

World Space vs Local Space

As a lot of people have already explained this concept I will just refer to this post for brevity:

https://medium.com/nerd-for-tech/local-space-vs-world-space-in-unity-6a9944470478