IOR Team Onboarding Reading List

1. Product Introduction:
   1. [SpatialOS](https://docs.improbable.io/reference/13.7/shared/concepts/spatialos) is a platform-as-a-service providing managed cloud services that solve common technical and production challenges of creating, iterating on, and hosting multiplayer games.
2. Architecture (e.g.thinking Spatially, Spatial as an instance of generalized ECS approach game and simulation design):
   1. What is ECS:
      1. <https://en.wikipedia.org/wiki/Entity_component_system>
      2. <https://github.com/junkdog/artemis-odb/wiki/Introduction-to-Entity-Systems>
      3. ECS in Unity: <https://unity3d.com/learn/tutorials/topics/scripting/ecs-overview>
3. Spatial’s version of ECS: Entities, Components, Workers, and Snapshots
   1. Core concepts of Spatial design: the building blocks of a Spatial project
      1. <https://docs.improbable.io/reference/13.7/shared/concepts/spatialos>
      2. <https://docs.improbable.io/reference/13.7/shared/concepts/world-entities-components>
      3. <https://docs.improbable.io/reference/13.7/shared/concepts/workers-load-balancing>
      4. <https://docs.improbable.io/reference/13.7/shared/concepts/interest-authority>
      5. <https://docs.improbable.io/reference/13.7/shared/concepts/layers>
      6. <https://docs.improbable.io/reference/13.7/shared/concepts/schema>
   2. Deep dive on what is a worker? (in short, a process that handles simulation for a subset of components for a section of the world - although that section could be the whole world)
      1. <https://docs.improbable.io/reference/13.7/shared/design/design-workers>
   3. SpatialOS project structure: Setting up a project
      1. <https://docs.improbable.io/reference/13.7/shared/project-layout/files-and-directories#layout-of-a-spatialos-project>
      2. In short, because the doc above is a little confusing:
         1. Every SpatialOS project folder must have 4 features. The spatialos CLI looks for these features to build and deploy projects.
         2. To build and deploy a project run the following commands specified here.
            1. <https://docs.improbable.io/reference/13.7/shared/operate/snapshots>
         3. Expected features of a project are:
            1. Schema directory

Folder that holds files defining components included in the world (like a health component defining current health on all entities, for example)

* + - * 1. Workers directory

Folder containing scripts (workers) that will be deployed and load balanced throughout the world to handle computation affecting entities

* + - * 1. [Spatialos.json](https://docs.improbable.io/reference/13.7/shared/project-layout/spatialos-json)

High level file containing project metadata

* + - * 1. [Launch json file](https://docs.improbable.io/reference/13.7/shared/project-layout/launch-config)

Launch configuration telling Spatial how many of each type of worker to start with a new project, and how to load balance them

* + - * 1. A snapshots folder, and file

File containing the initial world state, defining existing entities and attached component. This is usually built with a script that is written to produce a snapshot file

<https://docs.improbable.io/reference/13.7/shared/operate/snapshots#creating-snapshots-from-scratch>

Snapshots and launching deployments:

<https://docs.improbable.io/reference/13.7/shared/operate/snapshots>

* + 1. Example directory structure: <https://github.com/spatialos/gdk-for-unity-fps-starter-project>
  1. The schema folder: Defining components using SpatialOS schemalang (Note: some of the content on commands and events will make less sense until reading the worker section):
     1. <https://docs.improbable.io/reference/13.7/shared/schema/introduction>
     2. <https://docs.improbable.io/reference/13.7/shared/concepts/schema>
     3. <https://docs.improbable.io/reference/13.7/shared/design/design-components>
     4. <https://docs.improbable.io/reference/13.7/shared/schema/reference>
  2. The workers folder: building workers to enact behavior on entities
     1. Review: What is a worker (re-included for relevance to this section): <https://docs.improbable.io/reference/13.7/shared/concepts/world-entities-components>
     2. Load balancing workers: <https://docs.improbable.io/reference/13.7/shared/concepts/workers-load-balancing>
     3. [Designing workers (including notes on worker directory structure):](https://docs.improbable.io/reference/13.7/shared/design/design-workers)
     4. Worker authority (defining what specific worker layer can see and change in the world):
        1. What are in interest (in short, interest is the subset of components a worker is interested in, has read access to, wants to receive updates about; authority refers to a single worker having write access over components):
           1. <https://docs.improbable.io/reference/13.7/shared/concepts/interest-authority>
           2. <https://docs.improbable.io/reference/13.7/shared/design/understanding-access>
        2. Defining interest and authority:
           1. Both authority and interest can be defined within a snapshot, via Entity ACL: <https://docs.improbable.io/reference/13.7/shared/schema/standard-schema-library#entityacl-required>
           2. Interest can also be established through query based interest, a more granular method for establishing interest in specific entities:

<https://docs.improbable.io/reference/13.7/shared/worker-configuration/query-based-interest#query-based-interest-beta>

* + - 1. Scripting behavior (how to write behaviors within workers and making changes to the SpatialOS world)

Telling Spatial to update a component (for components you are authoritative over):

<https://docs.improbable.io/reference/13.7/cppsdk/using/sending-data#sending-data-to-spatialos>

Telling Spatial to create or destroy an entity (after a deployment has started):

<https://docs.improbable.io/reference/13.7/shared/design/commands#creating-and-deleting-entities>

Receiving and responding to updates for components (you’ll receive component updates for components you have read access over)

<https://docs.improbable.io/reference/13.7/shared/design/operations#the-dispatcher-and-the-view>

Two other message types: (1) Commands: implementing behavior on one specific entity:

<https://docs.improbable.io/reference/13.7/shared/design/component-best-practices#when-to-use-commands>

<https://docs.improbable.io/reference/13.7/shared/design/commands#commands>

Two other message types: (2) Events: Communicate with all entities that have a component:

<https://docs.improbable.io/reference/13.7/shared/design/component-best-practices#when-to-use-events>

* + 1. Tools and coding languages for writing workers
       1. Overview:<https://docs.improbable.io/reference/13.7/shared/dev-tools-intro>
       2. Using the Unity Gdk (recommend the FPS starter project feature walkthrough): <https://docs.improbable.io/unity/alpha/welcome>
       3. Language workers (low level sdks allowing workers to be written in programming languages)
          1. C++:

<https://docs.improbable.io/reference/13.7/cppsdk/introduction>

Recommend experimenting with the starter project:

<https://github.com/spatialos/CppBlankProject>

Setting up a worker:

<https://docs.improbable.io/reference/13.7/cppsdk/building>

Boilerplate to establish SpatialOS connection and define components:

<https://docs.improbable.io/reference/13.7/cppsdk/using/connecting>

<https://docs.improbable.io/reference/13.7/cppsdk/using/providing-components>

Writing worker code (see worker section above for concept explanations):

Sending component updates:

<https://docs.improbable.io/reference/13.7/cppsdk/using/sending-data>

Processing operations:

<https://docs.improbable.io/reference/13.7/cppsdk/using/receiving-data>