**Note: Login with school account to use Visual Studio 2017 in lab, it’s possible to get Assembly-CSharp to replace Misc. Files after several tries**

**Goal: Shoot a raycast from KFP position to point where mouse clicked**

gathering useful APIs ("tools") here. Maybe best approach before starting to code is research APIs and concepts? Learning and stuff

1. if mouse button is clicked, get its position

2. a raycast needs to shoot from KFP's position and to the direction of the mouse click

Debug.DrawLine(Vector3 start, Vector3 end, Color color = Color.white, float duration = 0.0f, depthTest = false);

//depthTest just checks whether line should be obscured by other objects or not

**class LineRenderer**

LineRenderer.SetPosition(int index, Vector3 position);

//void, index determines which position to set and the position is the Vector to set

LineRenderer.SetPositions(Vector3[] positions);

//void, takes in an array of positions to set

**Camera.main.ScreenPointToRay**

//returns a ray in world space

ScreenPointToRay(Vector3 pos);

struct Ray - represents rays, infinite lines starting from an origin going to some direction

Constructor:

Ray(Vector3 origin, Vector3 direction)

Ray ray = new Ray(transform.position. transform.forward);

Vector3 GetPoint(float distance); //returns a point some distance along ray

**Input.mousePosition**

- returns a Vector3 of the position of the mouse in **pixel coordinates of screen**

(0,0 bottom left, up to screen width and screen height)

**Physics.Raycast - method returns a bool**

**Raycast(Vector3 origin, Vector3 direction, float maxDistance = Mathf.Infinity, int layerMask, QueryTriggerInteraction queryTriggerInteraction)**

***maxDistance*** Mathf.Infinity means the raycast goes on forever, checking for collisions

* A raycast can spawn from KFP, go to the **direction of the mouse click** for some distance
* Layer masks not needed right now

***QueryTriggerInteraction*** is an enum… seems to be used to check whether **triggers are hit by the raycast or not**

* Not needed right now

**RaycastHit**

**-** a struct used to get information from a Raycast

- to do this, an out parameter is used to retrieve a RaycastHit from a Raycast method

**Raycast(Vector3 origin, Vector3 direction, out RaycastHit hitInfo, float maxDistance, int layerMask)**

**RaycastHit.point** can be used to get the point in world space where the ray hit the collider

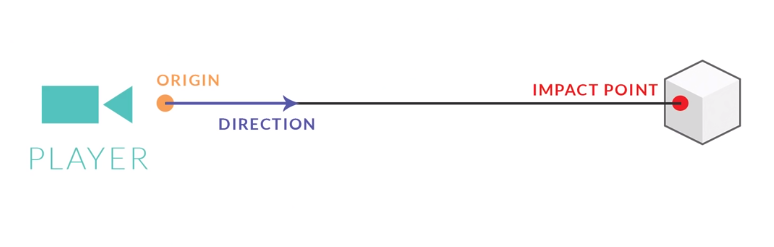
**RaycastHit.distance** gets the distance from the ray’s origin to the impact point

A raycast is just a method returning a bool

**Shooting in Unity uses Raycasts**

1. Make damage and range variable values

Raycasts return true if they hit something, and false if nothing was hit



RaycastHit hit;

If (Physics.Raycast(transform.position, **mouse’s click position**, out hit, range))

{

//at this point a raycast is shot, but how to get mouse’s click position?

}

//Input.mousePosition is in pixel screen coordinates, but the raycast needs to shoot in world space of the actual game world

**Public Vector3 Camera.ScreenToWorldPoint(Vector3 position)**

* Converts a screen space position to world space

Camera.main.ScreenToWorldPoint(Input.mousePosition);

**Goal: Render a raycast so it shows on screen**

Debug.DrawRay(Vector3 start, Vector3 dir, Color color = Color.white, float duration = 0.0f. bool depthTest = true);

Debug.DrawRay(transform.position, mousePos, Color.white, 1f, false);