unity

CHEAT SHEET

useful links:

Video Tutorials
https://unity3d.com/learn/tutorials/modules

MonoBehaviour Lifecycle
http://www.richardfine.co.uk/2012/10/unity3d-monobehaviour-lifecycle/

Enhanced Cheat Sheet
http://forum.unity3d.com/attachments/unitycheatsheet-pdf.17037/

Editor Shortcuts:

STRG/CMD + P = toggle play mode

STRG/CMD + SHIFT + P = toggle pause

STRG/CMD + D = duplicate selection

(when mouse is over scene view) **F** = move current view to selection

STRG/CMD + SHIFT + F = align selection to current view

STRG/CMD + LMB = snapping

(hold down) **V** = vertex snapping

Common Code Snippets:

Handling GameObject Lifetime

```
// creates a new copy of myPrefab at a position with a zero
rotation
GameObject newGO = GameObject.Instantiate(myPrefab,
this.transform.position, Quaternion.identity);

// destroys the GameObject newGO
GameObject.Destroy(newGO);

// destroys the GameObject newGO after 3 seconds
GameObject.Destroy(newGO, 3.0f);

// preserve the current GameObject when loading a new scene
GameObject.DontDestroyOnLoad(this.gameObject);
```

Finding Stuff

```
// returns a reference to a GameObject by name
GameObject theChair = GameObject.Find ("World/Room/Chair");

// returns a reference to a GameObject by tag
GameObject thePlayer = GameObject.FindGameObjectWithTag("player");

// returns a reference to a Component on same GameObject
SpriteRenderer mySpriteRenderer =
this.GetComponent<SpriteRenderer>();

// returns a reference to a Component on another GameObject
SpriteRenderer someSpriteRenderer =
anotherGameObject.GetComponent<SpriteRenderer>();
```

Getting User Input

```
// returns an Axis as float
float axisHorizontal = Input.GetAxis("Horizontal");

// returns true on Button down
// ("Jump" must be defined in Inputsettings)
bool buttonSpace = Input.GetButtonDown("Jump");

// returns true on Button down
bool keySpace = Input.GetKeyDown(KeyCode.Space);
```

```
// returns true on LeftMouseButton down
bool leftMouseBTN = Input.GetMouseButtonDown(0);

// returns mousePosition as Vector3
Vector3 mousePosition = Input.mousePosition;

// first: check if there are fingers on the screen
if(Input.touchCount > 0)
{
// returns position of first finger as Vector2
Vector2 fingerPosition = Input.touches[0].position;
}
```

Collision detection / Trigger detection

```
// OnCollisionEnter is called automatically by Unity
// if two objects collide with each other and both objects have
// a collider attached, and at least one of them also has a
rigidbody
void OnCollisionEnter(Collision collision)
{
Debug.Log(collision.gameObject.name);
}

// works exactly as above but with 2D colliders and 2d rigidbodies
void OnCollisionEnter2D(Collision2D collision)
{
Debug.Log(collision.gameObject.name);
}

// OnTriggerEnter is called automatically by Unity
// if two objects with colliders intersect and one collider is
// marked as trigger. Also at least one of them needs a rigidbody.
void OnTriggerEnter(Collider otherCollider)
{
Debug.Log (otherCollider.gameObject.name);
}

// works exactly as above but with 2D colliders and 2d rigidbodies
void OnTriggerEnter2D(Collider2D otherCollider)
{
Debug.Log (otherCollider.gameObject.name);
}

// there are also functions for OnCollisionStay, OnCollisionExit,
// OnTriggerStay, OnTriggerExit
```

Moving objects

```
// moving an object via setting its position directly
void Update()
{
// Time.deltatime is required in order to have a smooth transition
this.transform.position = this.transform.position + moveVector *
speed * Time.deltaTime;
}
```

```
// moving an object via physics velocity
// Time.deltatime no required because velocity considers time
already
this.rigidbody.velocity = moveVector * speed;
```

Math functions

```
// Clamp returns someValue if its between min and max, otherwise
it
// returns min or max
Mathf.Clamp(someValue, min, max);
```

```
// Abs returns always absolute (positive) value
Mathf.Abs(-25.3f); // returns 25.3f
```

```
// Lerp returns a value between from and to according to progress
// progress needs to be between 0 and 1 (0 = from, 1 = to)
Mathf.Lerp (min, max, progress);
Vector3.Lerp(from, to, progress);
Color.Lerp (from, to, progress);
```

Random

```
// returns a random number between 0.0 (incl.) and 1.0 (incl.)
float random = Random.value;
```

```
// returns a random float between and minF (incl.) and maxF
(incl.)
float random = Random.Range(minF, maxF);
```

```
// returns a random integer between and minI (incl.) and maxI
(excl.)
int random = Random.Range(minI, maxI);
```

Debugging

```
// print to the console
Debug.Log("lorem ipsum");
print("hello world");

// print a warning to console
Debug.LogWarning("This is a warning!");

// print an error to console
Debug.LogError("Ooops, an error occured!");
```

Time

```
// Delta time since the last frame
Time.deltaTime

// Current time (since game was started) in seconds
Time.time
```