Digitalt skapande - Vårtermin - Namn:

Kopiera delarna nedanför för varje lektionstillfälle. Skriv det senaste inlägget överst så att du alltid har det senaste längst upp. Ibland används inte alla rubriker vilket är okej.

Exempel: spara alltid en tom mall längst upp

Dag: **Vecka**: **Kurs**: Digitalt Skapande **Lärare**: Timmy

Lektionsmål:

Lektionsaktiviteter/lektionsuppgifter:

Lektionsanteckningar:

Lektionsutvärdering:

Hemuppgift:

Tema: Eget projektarbete

Länk till procjetet på github är :

https://github.com/nooaandersson/DigGame

Bild på hela koden för dokumentation purps:

https://carbon.now.sh/?bg=rgba%28171%2C+184%2C+195%2C+1%29&t=night-owl&wt=none&l=javascript&ds=true&dsyoff=20px&dsblur=68px&wc=true&wa=true&pv=56px&ph=56px&ln=false&fl=1&fm=Hack&fs=14px&lh=133%25&si=false&es=4x&wm=false&code=%253Chtml%253E%250A%2520%2520%253Chead%253E%250A%2520%2520%2520%2520%253Cscript%2520src%253D%2522https%253A%252F%252Fcdnjs.cloudflare.com%252Fajax%252Flibs%252Fthree.js%252Fr124%252Fthree.min.js%2522%253E%253C%252Fscript%253E%250A%2520%2520%253C%252Fhead%253E%250A%2520%2520%253Cbody%253E%250A%2520%2520%2520%2520%253Cscript%2520src%253D%2522https%253A%252F%252Fcdnjs.cloudflare.com%252Fajax%252Flibs%252Fthree.js%252F88%252Fthree.min.js%2522%253E%253C%252Fscript%253E%250A%2520%2520%2520%2520%253Cscript%2520src%253D%2522https%253A%252F%252Fthreejs.org%252Fexamples%252Fjs%252Fcontrols%252FTrackballControls.js%2522%253E%253C%252Fscript%253E%250A%2520%2520%2520%2520%253Cscript%253E%250A%2520%2520%2520%2520%2520%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%252

0var%2520script%2520%253D%2520document.createElement%28%2522script%2522%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520script.onload%2520%253D%2520function%2520%28%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520var%2520stats%2520%253D%2520new%2520Stats%28%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520document.body.appendChild%28stats.dom%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520requestAnimationFrame%28function%2520loop%28%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520stats.update%28%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520requestAnimationFrame%28loop%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%257D%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%257D%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520script.src%2520%253D%2520%2522%252F%252Fmrdoob.github.io%252Fstats.js%252Fbuild%252Fstats.min.js%2522%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520document.head.appendChild%28script%29%253B%250A%2520%2520%2520%2520%2520%2520%257D%250A%2520%2520%2520%2520%2520%2520%2520%252F**%250A%2520%2520%2520%2520%2520%2520%2520*%2520Generate%2520a%2520scene%2520object%2520with%2520a%2520background%2520color%250A%2520%2520%2520%2520%2520%2520%2520**%252F%250A%2520%2520%2520%2520%2520%2520var%2520getPoints%2520%253D%2520true%253B%250A%2520%2520%2520%2520%2520%2520var%2520gravetyOn%2520%253D%2520true%253B%250A%2520%2520%2520%2520%2520%2520var%2520pointsText%2520%253D%25200%253B%250A%2520%2520%2520%2520%2520%2520var%2520points%2520%253D%2520%255B%255D%253B%250A%2520%2520%2520%2520%2520%2520var%2520coordsx%2520%253D%2520%255B%255D%253B%250A%2520%2520%2520%2520%2520%2520var%2520coordsz%2520%253D%2520%255B%255D%253B%250A%2520%2520%2520%2520%2520%2520function%2520getScene%28%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520var%2520scene%2520%253D%2520new%2520THREE.Scene%28%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520scene.background%2520%253D%2520new%2520THREE.Color%280x111111%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520return%2520scene%253B%250A%2520%2520%2520%2520%2520%2520%257D%250A%2520%2520%2520%2520%2520%2520function%2520Tp%28x%252C%2520y%252C%2520z%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520sphereGroup.position.set%28x%252C%2520y%252C%2520z%29%253B%250A%2520%2520%2520%2520%2520%2520%257D%250A%250A%2520%2520%2520%2520%2520%2520function%2520getGravetyOn%28%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520gravetyOn%2520%253D%2520true%253B%250A%2520%2520%2520%2520%2520%2520%257D%250A%2520%2520%2520%2520%2520%2520function%2520getGravetyOff%28%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520gravetyOn%2520%253D%2520false%253B%250A%2520%2520%2520%2520%2520%2520%257D%250A%2520%2520%2520%2520%2520%2520function%2520BackTrackLog%28%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520console.log%28%2522something%2520is%2520undefined%2520in%2520RenderNodes%2522%29%253B%250A%2520%2520%2520%2520%2520%2520%257D%250A%250A%2520%2520%2520%2520%2520%2520function%2520renderNodes%28scene%252C%2520x%252C%2520y%252C%2520z%252C%2520namee%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520if%2520%28namee%2520%253D%253D%2520undefined%29%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520BackTrackLog%28%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%257D%2520else%2520%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520var%2520geometry%2520%253D%2520

new%2520THREE.CubeGeometry%281%252C%25201%252C%25201%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520var%2520material%2520%253D%2520new%2520THREE.MeshPhongMaterial%28%257B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520color%253A%25200xd6921d%252C%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520emissive%253A%25200xaf752a%252C%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520side%253A%2520THREE.FrontSide%252C%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520flatShading%253A%2520true%252C%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%257D%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520var%2520node%2520%253D%2520new%2520THREE.Mesh%28geometry%252C%2520material%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520node.name%2520%253D%2520namee.toString%28%29%253B%250A%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520%2520node.position.set%28x%2520*%2520100%252C%2520y%252C%2520z%2520*%252010

Well länken ser lite funky ut men den leder till en bild av koden för den var för stor för att läsa bilden kommer ligga nedan för content så ja... :

Dag Trosdag vecak 7 kurs digdig lärare Timmy

Koden nedan visar hur man renderar poängen. Som du kan se av x,y,z så är det kordinater på min array render system som jag har gått igenom nedan. Systement ger mig värden som varje poäng ska renderar efter. De här poängen får sendan en collsion box som raknas ut relativt till var spelaren är. Om spelaren inte är nära så kommer boxen inte att renderas. Detta hjälper med FPS då vi inte behöver render saker som man inte kommer se ändå men måste vara där om spelarn är att kollidera med objectet. Koden för kollsion kommer nedan med en mer spesefic beskriving. Men tillbaka till koden om renderingen av poängen så ser vi att vi are ett namee. Detta refferare till vad poängen ska ha för namn. För om den inte har ett namn så kommer jag inte kunna hitta den senare när den ska bli bortagen då spelaren kolliderar med poänget. Så objected för ett namen vilket reperosnerar av mitt render system vars array har en satt längd alltså hur många värden som finns i arrayen. Med de här värdena så kommer en for loop att gå igenom varje postion av arrayen och ge ett poäng en viss status. Allså som den poänget är en tid givare eller en poäng givare. Koden som loopar igenom detta finns under första bilden.

```
function renderNodes(scene, x, y, z, namee) {
  if (namee == undefined) {
    BackTrackLog();
  } else {
    var geometry = new THREE.CubeGeometry(1, 1, 1);
    var material = new THREE.MeshPhongMaterial({
      color: 0xd6921d,
      emissive: 0xaf752a,
      side: THREE.FrontSide,
      flatShading: true,
    });
    var node = new THREE.Mesh(geometry, material);
    node.name = namee.toString();
    node.position.set(x * 100, y, z * 100);
    console.log(coordsz);
    console.log(node.name);
    scene.add(node);
  }
}
```

```
function RenderPointsArray(scene) {
  for (i = 0; i < coordsx.length; i++) {
    //Collison(scene, coordsx[i], -50, coordsz[i])
    renderNodes(scene, coordsx[i], -45, coordsz[i], i);
  }
}
```

Spelet ser för tilfället ganska dåligt ut men mening och den är viktig är att jag inte gör spelet för att göra det snyggt eller att de ska se bra ut. Jag focuserar på att koda spelet. Alltså the "art off code" viket är en vikigt sak om programming att man ska lära sig nya saker hela tiden. Så med hjälp av detta projectet så har jag lärt mig hur man gör egen collision dettection hur man bygger upp en egen render engine men också hur man räknar ut raycast system och mycket annat. Spelet ska ha en mening annans är det ingen spel. För tillfället är meningen med spelet att man ska klara det på så kort tid som möjligt. Om vi får chancen sen att bygga vidare på projected mot slutet av terminen så kommer jag att besöka det

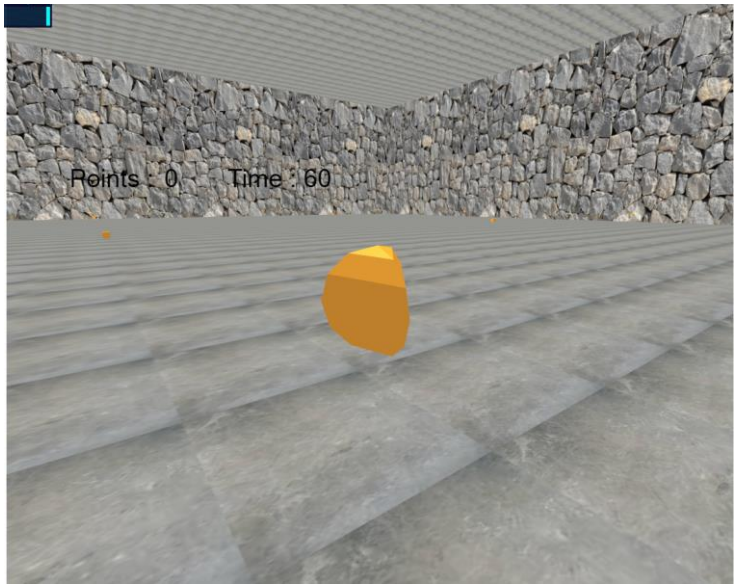
här projected igen och enbart bygga upp det för att träna en AI och se hur långt tid det tar för en AI att hitta den bästa stragein för att klar spelet snabbast. För i slutändan så kommer det bara finnas en path som man kan tar för att klara spelet snabbast då poängen inte ändrar postion eller ändrar form. En bild på spelet kan du se nedan. I bilden nedan kan du se hur de oranga kuberna repersneratr poäng sen när du spelar spelet så kommer du kunna se att de ger dig 7 päng att äta en av kubena varför det är så är för att jag vill att man ska nå höga pääng för då kan du bli fångad av scoren. Sen finns det en annan anleding vilket är att jag kommer länga till andra poäng varianter som ger dig mer eller mindre poäng. Men detta kommer jag lägga till om när AI ska impelmeteras. Det som jag mer har gjort är att Timern nu funkar man har tid på sig att ta så många poäng som möjligt och då måste man hitta så många paths som möjligt för att kunna göra detta så effecktivt som möjligt. Det som ska göras innan inlämning är att sätta timer och points till en bättre positon mitt i skärmen känns lite funky. Sen ska jag lägga till olika färger på pääng en.

Det du kommer märka när du spelar det är att man måste vara ganska exakt på poängen för att få dom. Detta har jag gjort för att göra det svårare att spela om man hade fått poängen direkt när man nuddar dem så hade man kunna samla de snabbt men du har man en helt ny asspekt till spelet viket gör att man måste vara exakt på pääng en med mitten av kroppen för att få den. Tanken var att det skulle bli roligare då det ger mer svårighet till spelet.

Jag pratade om “Art of code” innan jag kommer länka till en yt video som går igenom “Art of code” men om jag ska sammanfatta det så är det hur man skriver koden och inte vad den blir. Utan arten är att koden funkar och att koden bygger upp något som sen kan bli bättre och att man lär sig nya saker. Du borde titta på videon den är extremt bra.

Länk till yt video “Art of code” : [The Art of Code - Dylan Beattie]


The Art of Code
Dylan Beattie
@dylanbeattie
NDC London 2020 #ndclondon
?



innan jag avslutar det här inlägget så kommer jag att gå igenom hur mitt collsion system funkar. Ur en mer avacnerav men ändå grundilig veration. Om du kollar på bilden nedan så är det koden som gör att du märker att min spelar har åkt in i poänget. Kollar man på den så ser den ganska simpel ut men när man bryter ner den i sina delar så är den ganska stor. Det koden börjar med är att gå igenom två arrayer med massa kordinater till olika postioner då mina poäng delar en array med alla andra object som ska renderas så måste man gå igeom massa saker. Det hade varigt enklare att gör en separat array för poängen men då hade jag fått ännu mer opimiseriong problem än vad jag redan har. Så Det koden faktisg gör är att den varje millsekunder går igenom hela arrayen och räknar ut om man är nära spelaren eller inte. Är man inte det så kommer collboxen inte att renderas som jag nämde innan. Men om spelaren är i närheten så kommer poänget att få en box vilket gör att spelaren kan kollideras med poängen alltså tjäna poänget. Detta bygger upp till en cool formel som du kan se på if statmenetsn. Här kollar den över if statmentsen och då gör den en uppfatting på postionen och namn. Sen om spelaren skulle kollidera med scen och spelar om det skulle kollidera med objektet så kommer den att ta bort objektet och säger det kommer den att spendera ett poäng i spelarens poäng text. Efter det så kommer att ta bort objektet får arrayen detta jag sen att

poänget inte kommer att renderas som tag bart igen. om poängen inte finns i min samling av världen så kommer den inte att kunna rendera sen och därför inte kunnat ta power från andra processer. detta gör då att man får ett bättre optimering jobb och man kan få högre fps.

```
function CollForPoints(scene) {
  for (i = 0; i < coordsx.length; i++) {
    if (Math.round(sphereGroup.position.x) == coordsx[i] * 100) {
      if (Math.round(sphereGroup.position.z) == coordsz[i] * 100) {
        console.log(coordsz);
        var GetObjectForRemoval = scene.getObjectByName(i.toString());
        scene.remove(GetObjectForRemoval);
        if(getPoints == true) {
          getPoints = false;
          pointsText = pointsText + 1;
          document.getElementById("points").innerText = pointsText.toString();

        }

      }
    }
  }
}
```

om vi nu kolla tillbaka på projektet och se om det var vackert skrivas som tas upp direkt eller om det var väl bättre att skriva något annat. Så skulle jag säga att det var mycket bättre att skriva det här projceted då jag lärde mig extremt mycket. Vissa av sakerna jag lärde mig hade jag aldrig att jag skulle skriva till exempel om hemsida eller om jag skulle skriva en softwere som räknade ut alla saker utan med hjälp av detta projektet så kommer jag kunna bygga vidare skapa andra spel skapas innovationer och många andra coola saker. Även om det har tagit många timmar för mig att skriva det här projektet och räkna ut om problemen som haft för det har funnits många problem där projektet några av sakerna har varit att jag inte har kunnat basen som jag skriver detta på en av de andra sakerna har varit att spelutveckling på hemsidor eller på webbplats är en jättestor skillnad från de sakerna är jag van vid alltså till exempel unity eller kanske unreal som jag också har varit lite med. Det saknar jag kunnat göra bättre nästa gång framförallt skriva mig eller boken de andra sakerna var att till exempel lägga mer tid på att animera spelare och animera så saker se bra ut istället för att koden bakom att bara skriva hade efter mer tid på detta projektet så hade jag absolut lagt ned tid på hur allting ser ut hur all miljö ska se ut och hur allting renderas på ett effektivt sätt. För en stor sak som jag har kommit fram till det är att när du kodar ett spel på hemsida så kommer det inte kunna

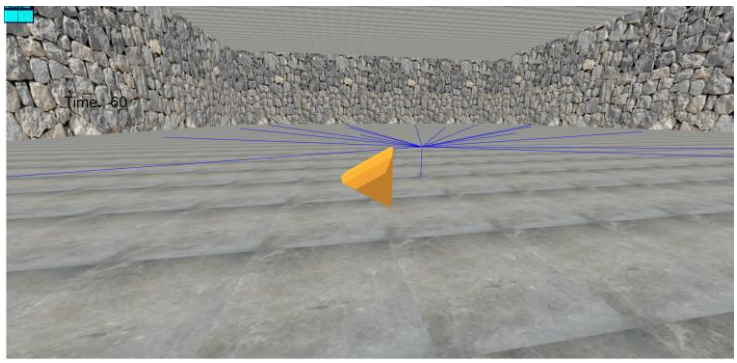
koda saker som är lika stora detta är ganska självklart så klart men jag tänkte inte att vara hur man hanterar några objekt så ska man tappa så mycket i fps. jag gjorde man ändå vilket var lite förvåning men det betydde att jag var tvungen att ändra hur jag kodade upp hela världen. detta blev en väldigt rolig men tidskrävande process som jag klart lärde mig extremt mycket av. Som man pratar om innan så kommer jag att se när är projektet om det är på den här lektionen eller om det är ett egen produkt så kommer jag att koda som är i kan lära sig hur man spelar spelet och hur man kan klara spelet snabbast självaste idén är att låta en dator lära sig spelet och sen så lära sig hur man klarade på snabbast tid eller hur man klarar det på så mycket poäng som möjligt eller till exempel hur man klarar sig för att hålla sig vid liv så långt in som möjligt de behöver dem ut lägga in andra saker som att poäng får en ny vespan alltså att de renderas på nytt vilket kan vara ganska tidskrävande men kan vara ganska roligt också.

En annan snabb sak som också är viktig att nämna att man kan skriva snabbt kort man kan skriva kropp som fungerar men är jättedålig och ingen annan kommer fatta den detta problem med haft innan då oftast mina projekt har passerat bara jag ska behöva läsa. Men under tiden som har gått så har jag kommit tillbaka till många gamla projekt och läste igenom det om jag kanske behöver någon slags del av det eller jag behöver någon annan sak från dig och då har jag kommit fram till att jag inte alls har kunnat förstå vad koden gör för någonting inte heller har dokumenterat någonting så att jag faktiskt vet hur någonting fungerar så detta också varit någon slags hjälpmedel i form av de här loggboken för att det ska kunna gå tillbaka till mig senare och faktiskt så att jag och min kommunikationsmetod gör detta min kollisions kan värdera detta men kan inte göra de här sakerna och det är extremt bra och dokumenterat då om man vill gå tillbaka ut på det och ändra saker så kan man hälla i. Det är även väldigt bra då förhoppningsvis i framtiden så kommer arbeta på såna här projekt fastigheter stöd och de måste projektet ha bra grund där man kan dokumentera mycket med fakta så att alla i gruppen förstår man ska göra. Får se när du kommer ut i jobb livet så kommer du bara arbeta i grupper det är oftast inte du själv som programmerare till projekt utan du är en gud av programmerare som programmerade och projektet och då måste man kunna

kommunicera dokumentera och veta vad alla andras kod gör framförallt min såklart. Det är detta jag kallar för art of code har begreppet är mycket mer än att man gör någonting liksom ska se ut på detta viset utan självaste utvisningen är det då skrivit det är det som du producerar i koden de många olika sätt att visa hur du kan klara någonting i kraft den här kursen där du kan rita och du kan filma en film gör allt möjligt. men alla sakernas gemensamma nämnare är att någonting digitalt som ser ganska coolt ut från bryta någonting och vill ha ett bra betyg på det så ska det vara bra ritat om du gör en film som inte har något budskap eller inte har de mening så kommer naturligtvis inte fått bra betyg på det samma sak gäller här skriver du en dålig kod som inte är en det är någonting som fungerar så kommer inte upp avtryck men om de lägger ner tid på koden så att koden fungerar då förhoppningsvis så kan man få ett bra betyg alltså traditionellt sätt så är det vad koden produserar som är det vikigast men i min vation så är det koden som är skapensen på pappret, filmen på tv och så linkade.

Dag måndag vecka 7 kurs digdig lärare timmy

Idag Har jag arbetat med ett collition system det är inte ett efectit system men det böjar bli klart. Jag har byggt det med vectorere bilden nedan visar hur vektornsna ser ut detta kommer man inte se in game sen men för stunden när jag räknar på de så måste jag ha de kvar. Vektorna är de sträken som är blåa. Som du ser så går varje sträk till en platfram. Detta har jag gjort med en array som håller all infomration om cordiknater till platforrmana. I mitt fall så räknar jag bara med enkla numer.



Mitt collsion system bygger på poinsts som renderas till varje node vilket är varje del av marken. Detta sätt låter mig bygga upp en stor variant till matematiken. Jag kommer använda mig av en fomel som heter “Math.floor” funktionen ger mig ett value som drar tillbaka mig till utgångspunkten.

Koden som det här poängsystemet kan man se på bilden nedan det enklaste sättet för mig att rendera koden är att bygga upp ett system som inte tar hänsyn till till de andra bitarna alltså andra planen som också vänder oss. Kunden som ser nedan gå igenom olika punkter i världen och efter det så men det gör den streck de här strecken representerar någonting som heter raycast ray kasten skicka ut en linje som kan visa på hur när var det spel ämnen som gör spelet i have cast funktion men jag har alltid velat göra en egen funktion vilket betyder att jag kommer rendera alla linjer med hjälp av olika vektorer sen så kommer jag räkna med att ta längden ifrån min spelare till längden ifrån strecken slutpunkt om längderna för kort så vet jag att okej då är det nära ett objekt som har en collsion applyer alltså collisions den. ja mycket bättre här system så är det viktigt att man egentligen inte renderar sträckor oftast i inrikes funktion så bygger man upp massa olika funktioner som lägger ut jättejättemånga streck om vi tar ett stort spel där rikaste väldigt stor del av spelet så vänder jag ut miljoner av linjer som är väldigt väldigt väldigt nära varandra när man då vet man exakt vad spelar han är du kan tänka dig som ett diagram när staplarna går

namn och namnen det finns jävligt sorterings algoritm som bygger upp detta jättebra.

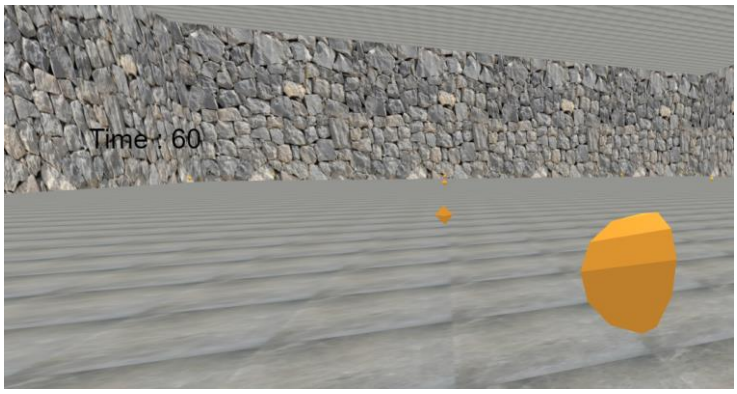
```
function Collison(scene, px,py,pz) {


  points.push(new THREE.Vector3(px * 100,py,pz * 100))
  points.push(new THREE.Vector3(sphereGroup.position.x, sphereGroup.position.y, sphereGroup.pos
  const geometry = new THREE.BufferGeometry().setFromPoints( points );
  const material = new THREE.LineBasicMaterial( { color: 0x0000ff } );
  const line = new THREE.Line(geometry, material)
  scene.add(line)
  return line

}
```

Om vi går in med koden så ser vi att den här kollektionen funktionen ta olika punkter från en grej som jag skapat den här grejen har alla olika koordinater till alla planen det ens en miljö där appen tar de här kommunerna flygplan rita ett streck som sedan kopplas ihop med spelarens position den här swea group är det spelaren. sen så går vidare till geometrin där har vi en enkel buffer och det är den som gör så att man vet var alla platfomrar ska vara. Det finns en array som gör detta. När spelet startar så räknar den ut var alla postioner är och utefter det så renderer den våra sträck eller raycast.

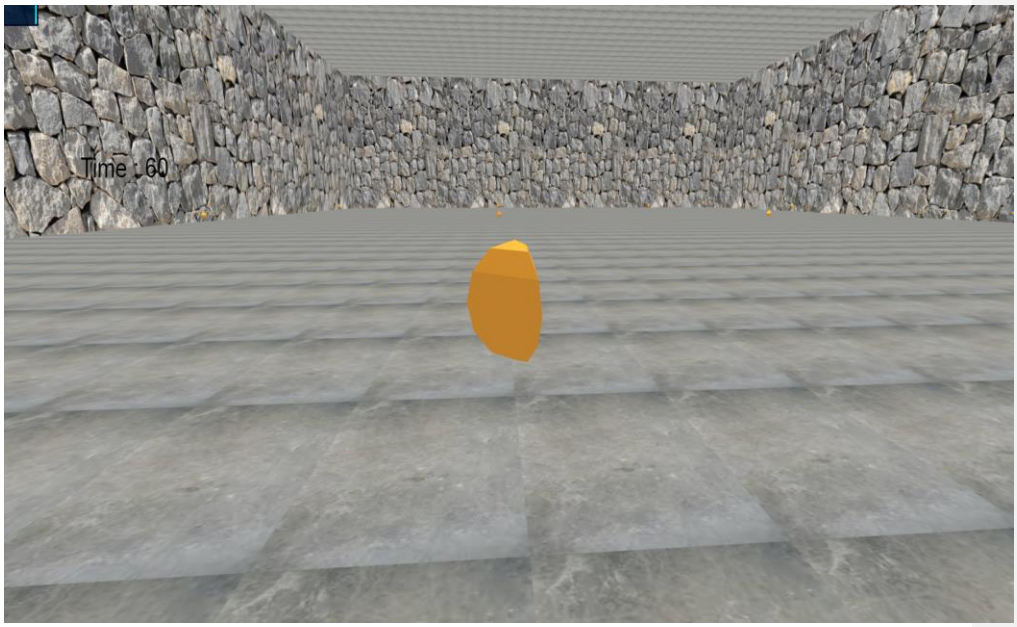
Jag har inte jättemycket tid kvar att utveckla det här spelet så jag kommer bygga upp ett system som göra att spelet är nästan klart. Det kommer finns en spelvariant typ som ett minigame. Och sen så kommer jag att utveckla spelet länge med längre tid. Spelet kommer gå ut på att samla så många poäng innan tiden är slut. Jag kommer spawna små bollar som man ska hämta för att få poäng.

Det som jag också har lagt in är en klocka som tickar ner för varje sekund tanken är att man ska ha på sig en minut för att plocka så många poäng som möjligt. Det som gör spelet roligt är att man måste hitta den bästa startegin för att hinna få så många poäng som möjligt. Jag tänker att man ska ha olika checkpoints som

gör att man får mer tid i framtiden. Men när jag är klar med det så kommer vi ha ett funkgerade spel som har ett mål. Bild på timern finns nedan:



Spelpoängen kan du se på bilden nedan.



Poängen erhåller samma fäg som spelare för tillfället den har även samma attributer som spelare bara att den är mindre. Det går snabbare att copy pasta min spelar koden en att skriva en ny som ändå kommer vara identisk till spelarens kod. Koden nedan visar hur man renderar poängen på spelplanen. Kodens rendersingprocess följer samma regler som raycast koden vilket gör att dens system blir likadant. Koden för poängen får alltså sinna kordinater från samma array postionnering. Koden nedan repersenterar detta.

Där den översta koden bestämmer vilka fäger och hur den ska se ut medans den under koden visar var objectet ska renderas alltså på vilka kordinater.

```
function renderNodes(scene, x,y,z) {
  var geometry = new THREE.SphereGeometry( 1, 1, 1 );
  var material = new THREE.MeshPhongMaterial({
    color: 0xd6921d,
    emissive: 0xaf752a,
    side: THREE.FrontSide,
    flatShading: true
  });
  var node = new THREE.Mesh( geometry, material );

  // create a group for translations and rotations
  var nodeGroup = new THREE.Group();
  nodeGroup.add(node)

  nodeGroup.position.set(x * 100, -45, z * 100);

  scene.add(nodeGroup);
  return [node, nodeGroup];
}
```

```
for(i = 0; i < coordsx.length; i++) {
  t = true;
  //Collison(scene, coordsx[i], -50, coordsz[i])
  renderNodes(scene, coordsx[i], -45, coordsz[i])
}
```

De som jag måste göra är att lägga till ett time system som räknar ner och se hur mycket poäng man har när tiden är ute. Sen måste jag även lägga till en poäng counter som hållar kolla på hur många pääng man har. Jag ska även lägga till collistion på mina poäng så att spelet vet när du colliderar med ett poäng eller inte. Detta gör att du kan veta när man måste rendera bort ett poäng från spelplan för att poänget redan har blivit taget. Sen om jag hinner så kommer jag lägga till att du få en ökad tid när du plockar upp vissa poäng men det gör jag bara om jag har tid i slutet. Då det tar extremt långtid att koda en spel motor då det är mycket matte och olika fromler som man måste göra för att allt ska funka så har projctet taget långt tid. Planen för pojectet var inte att göra ett trippel AAA game utan att lära mig en nu arrikteture och strukture vilket jag har lärt mig. Jag har gjort spel motorer innan fast aldrig direct till webben.

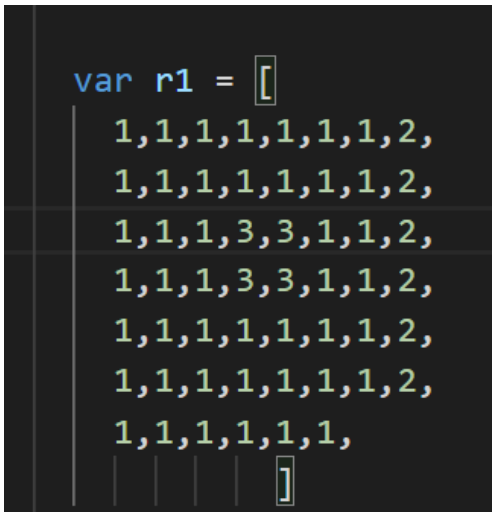
Tanken fårn börja var att göra ett spel som var mer RPG aktikgt men det jag har lärt mig är att det tar mycket längre tid än vad man tror. I de andra spel jag har gjort så har det alltid funnits inbyggda saker som kan hjälpa mig med t.ex. kollison i detta var jag tvungen att skriva mitt eget system viket klart tar längre tid. När jag har gjort spel i Unity så har jag alltid fastnat i en dvala som aldrig vill ta slut så jag blir trött på spelet då det inte blir någon utmaning när man redan vet exakt hur man ska göra. Det är roligare när man börjar fårn noll och sen bygger upp alla sakerna från grunden.

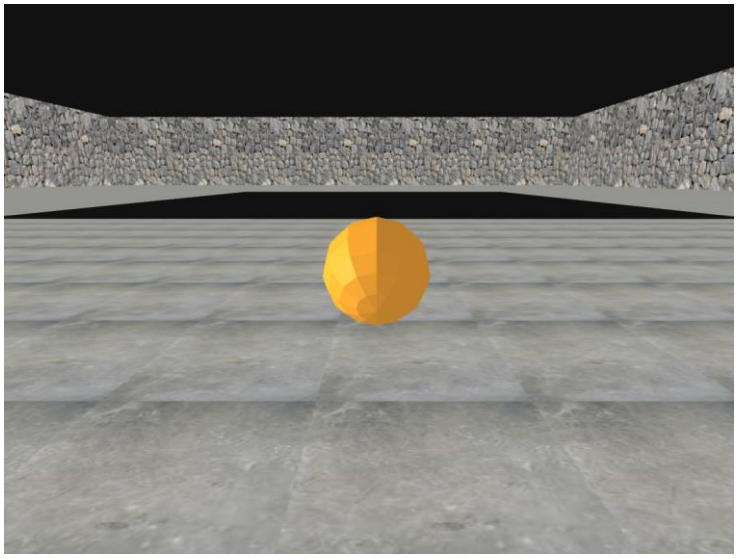
Till nästa lektion kommer jag göra en bättre timer och poäng räknare. Efter det så borde spelet fungerea som ett rikigt spel, alltså där det finns ett mål med spelet. För tillfället finns det inte det man är bara en boll som rullar runt på en spelplan viket inte är så kul.

Dag torsdag vecka 6 kurs digdig lärare Timmy.

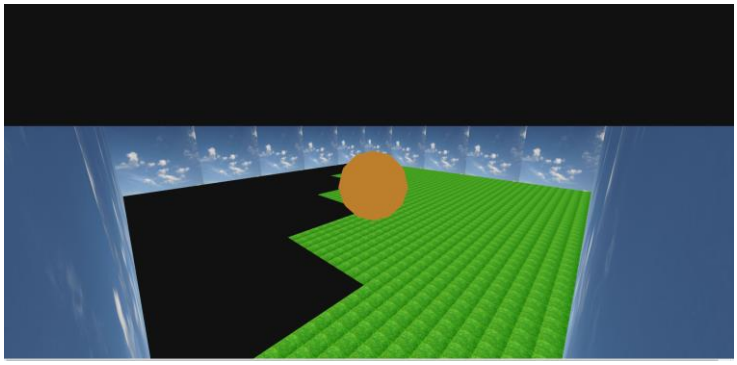
Idag har jag arbetar med olika saker jag har finslippat på mitt renderings system men också börjat skapa nya texturer.

Om vi börjar med det nya renderings systemetn så går det up olika saker. Första stora delen är att man skapar en array som håller några tal de talen kommer representera vår spelar plan. Om du kollar på bilden nedan så kan du se hur arrayen representerar välden. Det du ser är att när arryen har 3 or på en viss plats så kommer koden inte rendera någon mark detta gör det då enklet att byta ut mark för att ha en annan teräng.





När jag skrev koden till renderingsystement fick jag ett visst problem med min matematiska formel vilket gjorde att spelet genererade makten som på bilden. Man kan säga att vi tappade halva genereringen. Men faktisk så var det så att marken genererades spegelvänt fast på andra sidan väggen. Nu har jag ingen bild på det men jag tyket det var en kul grej. Finns alltid felkällor i programmering.



Jag spenderar av en stor tid idag för att kunna optimera texturere så det inte laggar. För nu när du laddar in första gången så kommer det lagga till lite för dig. Detta vara bara några sekunder sen så försvinner det och du får bra fps ingen. Men det är under de sek som man laddar in som det blir lite laggit detta berero på mina texturer. I och med att three js som jag använder för att koden spelet vill ha en viss upplösning på bilderna och jag ha andra upplösningar så kommer three js att behöva andra upplösningen när jag laddar in i spelet det är lite krävande att förnedra en upplösning på en bild i alla fall för en spelmotor.

En del av de matematiska fomelerna som jag kom fram till var (1000 * längden av min array) - 500 då kommer jag få ut en posstion på en grid. På den här griden kan jag sen rendera vad som hälst. Med den här matte formel så är det bara mark texturer som jag kan rendera detta beror på att planen som utgör marken är gjord av 1000 pixlar styck det är därför vi säger att vi ska gångrea mängden av min array med 1000 för om längden är 2 så kommer plat blir plaserat på postion 2000 vilket är en ifrån första raden. Varför vi sen tar och tar bort 500 är för att planen inte är kuber alltså de där inte 1000 tjock. Så om man inte tar bort 500 så kommer planets sidor att inte passa in med de andra alla måste samstämma för att få en geometrisk figur. Jag hade också antenknat i min kod och såg att jag anteckant att jag hade en till felkälla när jag skrev detta det var att i och med att vi inte har static tal så kommer vi alltid ha en liten förening på grund av min gravitation funktion som dra allt neråt. Funktionen är ganska enkelt den kommer jag bifoga nedanför.

```
function gravety() {
  if(sphereGroup.position.y > -450) {
    sphereGroup.position.y -= 2.5
  }


}
```

Funktionen för gravitation kommer att uppdtareas och blir mer komplex desto mer saker som jag lägger in för tillfället så kollar jag bara igenom min funktion som håller spelaren och drar den den med en kraft på 2.5 pixlar per milisktund detta betyder att man inte får en så snabb nergångn och man axxeleererar inte desto högre upp man är detta är ett mål att fixa det sen. Jag måste även lägga till en animation på min jump funktion vilket är den funktionen som gör att du kan hoppa för tillfället som du såg i måndag så tpas man bara upp i luften och sen så faller man ner med gravitaionen.

Mål för näsa vecka är att påbörja animationen men också lägga till fler saker som gör det enklet för mig att skapa en kol spelupplevelse då menar jag att rendera in olika fineder och object som man kan kollidera med. Även att utöka min kollitions funktion så man kan kollidera med alla saker i välden.

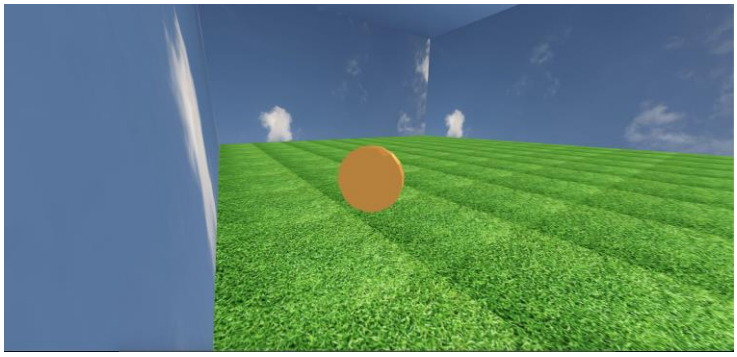
Dag Måndag vecka 6 kurs digdig lärare Timmy.

Idag har jag valt att byta renderings process jag hade innan en plan lösning vilket byggdes av attt jag kodaden in var olika plan skulle vara detta gjorde att man inte har likee mycket kontroll över var man sätter ut de olika plan detta gör att man blir begränsade till en planlösning som gör att du bara renderar delar äv en värld. Detta är ett problem som gör att du blir begränsad. Så mitt nya sätt är att sakpa en array : en array är en samling av värden som hålls i en varablen ett exemple är att E = [1,0,0,0,1] detta är enn array som hålla värdena 1,0,0,0,1 jag kommer använda denna metoden att för då kan jag sätta att en sak har ett värde i arrayen så om vi säger att 1 repersentterar en box så kommer man kuna rendera boxar på alla värdeen som ärr 1 i arrayen. Kollar vi tillbaka till mitt exemple så ser vi att vi kommerr ha två boxarr i varje hörn. Man kan säga att array bygger upp spelplanen för spelet. Så om vi bygger en array som har 2 med värde vägg så kommer vi kunna sätta ut massa väggar runt om spelplan med en enkel modifingena av arrayen.

Tanken med detta är att vi ska kunna bygga en spelplan som man kan interagera med enkelt det skapar även möjligheter för modifingen ingame vilket betyder att du kommer kunna bygga egna banor och t.ex. bygga hus och annat utan att det blir för jobbitt att rendera det. Det finns även andara möjlighete som t.ex. optimisereing man kommer inte lagga lika mycket då man alltid vet var spelarn är någonstans så kommer man inte behöva reendera innan alla väggar direkt utan man kan rendera det utifrån var spelaren är på spelplanen.

```
function getBackground(scene, loader) {
  var imagePrefix = '';
  var directions  = ['right', 'left', 'top', 'bottom', 'front', 'back'];
  var imageSuffix = '.bmp';
  var geometry = new THREE.BoxGeometry( 1000, 1000, 1000 );
  var texture = loader.load("sky.jpeg")
  var material = new THREE.MeshBasicMaterial({map:texture, side: THREE.BackSide})
  var materialArray = [];
  var gameArray =
  [
    1,1,1,1,
    1,0,0,1,
    1,0,0,1,
    1,1,1,1,
  ]

  var sky = new THREE.Mesh( geometry, material );
  scene.add(sky);
}
```

Här har vi funktion som kommer rendera alla väggar och object runt om i världen den är inte klar men du kan se en repersnation av arrayen. Kollar man i speleet så kommre man kunna se att vi har 4 väggar som då reprenseerar av 1or.



Som du kan se så är våra väggar de som har en sky texture. Nu kan man inte se alla fyra men bakom kamran så finns det också en vägg med sky texture. Detta sättet gör att man sen kan välja att inte rendera väggen som är bakom i och med att spelarn ändå inte ser den väggen. Detta hjälper att göra opttimiseringen av spelet och gör då att vi kan ha mer saker i spelet men motarebetar lag i fråm av fps lag. Den här prosecen bygger på en kontrollerad miljö som bygger upp sig av både spelare och utveklare. Man kan enkelt modifiera terräning till olika saker och pågrund av det så kommer spelarna få en bättre upplevelse. Spelarna kommer även enkelet kunna köpa sina egna hus och object för att modifiera sitt ställe. Men då kommer man behöva pengar vilket är nästa del i utvklingen. Alltså att man ska kunna på något sätt tjäna pengar.

Men tanke kring ekonomin kommer vara byggd runt att man dödar monster så får man olika saker som man se kan sälja. Detta är däremot tidfördrivande detta gör att vi i början bara kommer ha att man tjänar pengar på att döda monster alltså att de inte droppar någon loot. Ett enkelt sätt att göra detta är att ha en databas som håller kolla på hur mycket du tjäna. Detta blir enklast då vi inte vill att spelarna ska kunna fuska till sig pengar med hjälp av kommande. Så om vi spara spelarnas pengar i en databas som kommer ingen spelarna kunna ändra sina pengar utan att jag antingen har skrivet fel i koden så att det finns möjligheter för en spelar att ändra sina pengar eller så kommer de att på något annat sätt få tag i pengar. Men om man kollar över alla system som finns så är en databas det mest säkra alternativet.

En databas är? : en databas är en server som håller data lite som array som jag pratade om innan så håller de värden som är satta. Så om databasen sätter att jag som spelar har 100 gold i mitt account så kommer värdet av arrayen vara 100 guld. Men i en database så sparas allt på serven så att ingen som inte har kontroll över serven har återkomst till datan som sparas. Utan det är bara jag som utvecklar som kan ändra några pengar.

Målet för dagen var att bygga upp ett render system som enkelt kan bli modifierat till det man vill. Sen var det även att komma på en bra ide för att hålla kvar spelar

i spelet så de kan spendera mer timmar i det. Det jag kom fram till var att bygga upp ett ekonomisystem som gör att spelar blir fångade i en grind.

Målet för nästa lektion men också hemuppgift är att skapa en bättre version av det nya renderings systemet. Så att man har fler object att sätta ut men också så dett är så optimerat som möjligt.

Aktiviteten för dagen var att få systemet att funka vilket jag gjorde.

Det som också ska namnas är skyboxen och hur den fungerar. Skyboxen är generarde till en texture som vi kan se på bilden så är texturen på skyboxen en satt statisc bild som inte ändras detta betyder att om man har fel persspektiv så kommer spelet se väldigt statict ut alltså att det tjänns som om man inte rör sig över huvud taget detta vill jag ta bort ett eneklt sätt är att flytta skyboxen längre ut. Så man inte blir lika focuserad på box vilket gör att det käns mer levande. Sättet är väldigt enklet så det finns chans att förbättra upplevelsen med andra knep som t.ex. ändra bilderna så att de följer en annan tån eller att de är mer dynamiska. En annan vikigt sak är att jag måste få bort de delar som gör att det ser ut som en kub alltså att man ser kantterna på planen. Jag har hittaten en bra vidoe om det. Länk : Coding Minecraft in One Week - C++/OpenGL Programming Challenge

I en del av videeon så går han igenom hur man gör n skybox bätttre fast detta har han skrivit i ett annat programmeings språk som heter c++.

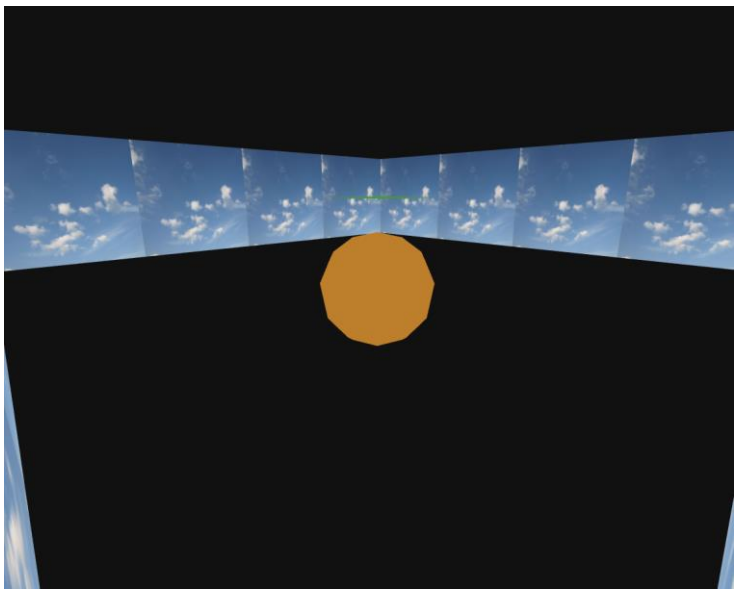
Jag har nu lagt till funktionen för att skapa större väggar och rendera en större game field koden är ganska enkel jag går igen. For loopsen fårn bildern det finns 4 forloops de reperonderat de fyra sidorna på min kub exlusin ovan och botten. Så vi rendera fyra sidor med den koden. Om ndu kollar på arryan som jag har ovan alltså r,l,u,b.

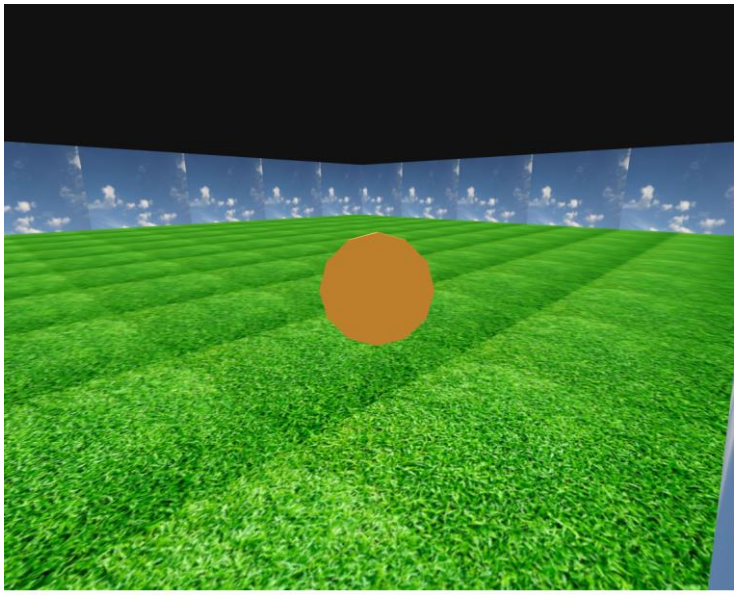
```
var r = [1,1,1,1,]
var l = [1,1,1,1,]
var u = [1,1,1,1,]
var b = [1,1,1,1,]
var imagePrefix = '';
var directions  = ['right', 'left', 'top', 'bottom', 'front', 'back'];
var imageSuffix = '.bmp';

var texture = loader.load("sky.jpeg")
for(iu = 0; iu < r.length; iu++) {
  var geometry = new THREE.PlaneGeometry( 1000, 1000, 1000 );
  var material = new THREE.MeshBasicMaterial({map:texture, side: THREE.DoubleSide})
  var skyr = new THREE.Mesh( geometry, material );

  scene.add(skyr);
  skyr.position.x = 1000 * iu;
  skyr.position.z = 875 * u.length
}for(ib = 0; ib < l.length; ib++) {

  var geometry = new THREE.PlaneGeometry( 1000, 1000, 1000 );
  var material = new THREE.MeshBasicMaterial({map:texture, side: THREE.DoubleSide})
  var skyl = new THREE.Mesh( geometry, material );

  scene.add(skyl);

  skyl.rotation.y = Math.PI / 2
  skyl.position.z = 1000 * ib;
  skyl.position.x = 875 * u.length

}

for(ir = 0; ir < r.length; ir++) {
  var geometry = new THREE.PlaneGeometry( 1000, 1000, 1000 );
  var material = new THREE.MeshBasicMaterial({map:texture, side: THREE.DoubleSide})
  var skyr = new THREE.Mesh( geometry, material );

  scene.add(skyr);
  skyr.position.x = 1000 * ir;
  skyr.position.z = -500
}for(il = 0; il < l.length; il++) {

  var geometry = new THREE.PlaneGeometry( 1000, 1000, 1000 );
  var material = new THREE.MeshBasicMaterial({map:texture, side: THREE.DoubleSide})
  var skyl = new THREE.Mesh( geometry, material );

  scene.add(skyl);

  skyl.rotation.y = Math.PI / 2
  skyl.position.z = 1000 * il;
  skyl.position.x = -500

}
```

Så om man ändra arrayn till 5 ettor istället för fyra som det är nu så kommer sidorna bli en längre. Skillanden kan du se på bilderna nedan.

Första bilden är när arrayn är 4 ettor och andra bilden är när array 5 ettor.

Så om du räknar kvadraterna som har en sky texture så kommer du att se att vi har fler kvardrater i andra bilden den renderas genom fler ettor i arrayn. Detta är extremt effektivt då vi kan rendera hur stor spelar yta som möjligt. Varför vi gör detta är för att vi ska kunna rendera så





Målen för nästa lektion har jag skrivit innan. 😊

Länk till project : https://github.com/nooaandersson/DigGame/tree/Sigge

Dag torsdag vecka 5 kurs digdig lärare Timmy

Målet för idag var att programmera så det ör det jag har haft fult fokus på min backlogg ät ganska stor så jag hinner inte skriva jätte mycket i min loggbok jag kommer göra det wn dag r jag inte har så stor bacjlogg då ska vi även gå lite ljuppare in på programmering och libet three.js men ockås socket js för mulitiplayer supporten. Annars har det varigt en bra dag vill du kolla in projectet så finns det på min github https://Gighub.com/nooaandersson/

Dag: Måndag **Vecka**: 5 **Kurs**: Digitalt Skapande **Lärare**: Timmy

Förra lektionen blev inte sparad. Men jag lärde sigge hur man anävnde gihub vilket var vikigt för oss då vi behöver lägga I hop vår kod sen. Men nu har han hoppat av projectet vilket gör den lektionen till en dålig tidsplan.

Lektionen 01/02 - 2021

Idag kommer jag att programmera.

Delmålet för idag är att få en camera att funka som kommer följa spelaren.

Dag: Måndag **Vecka**: 4 **Kurs**: Digitalt Skapande **Lärare**: Timmy

Vi har bytat project programmeringen är likadan och kommer inte att ändras

Som ett exemepl så kommer vi att skriva ett game som är likande https://www.kodeclubs.com/ detta spelet fast lite mer medivel och med vapen och så. Detta kommer klart att ta ett tag men I och med att vi är två så kommer vi nog hinna ha en prototyp klar tills inlämningen sker.

Idag ska jag arbeta med github och hur det fungare det kommer vara våran code base där vi kan lägga upp hur och vad som vi ska arbeta med. Vi kommer dedikera den här lektion till att lära sigge hur man använder github då det är där vi kommer lägga upp koden så alla kan ta del av den.

Mål för dagen: lära sigge använda GITHUB : https://github.com/

Update: det finns mycket att göra på github men vi har fått sigges github desktop att fungera nu ska vi bara koppla ihop det med hans account och project så borde allt funka. https://desktop.github.com/

Kommenterad [A1]: Vad tänker ni rent vissuellt. Exemplet har väldigt högt produktionsvärde när det kommer till animeringar och 3d-modeller. Vad tänker ni att ni ska göra? :)

Kommenterad [A2]: Länka gärna direkt till ert projekt påm github

[redacted]

Dag: Torsdag **Vecka**: 3 **Kurs**: Digitalt Skapande **Lärare**: Timmy

Lektionsmål: Skriv din projektplan.

Lektionsaktiviteter/lektionsuppgifter:

Du ska planera ditt arbete i förväg och hålla dig till planen i så stor mån som möjligt. När du skriver din tidsplan på torsdag v.3 ska du bygga ett veckoschema med delmål. Tanken är sedan att du ska redovisa varje lektion hur det går i ditt arbete. Under rubriken ”lektionsmål” skriver du in ditt eget delmål för varje lektion efter v.3

Lektionsanteckningar:

skriva vad jag ska göra kort # ,

läsa på om socket och hur det funkar # ,

skriva planeringen #,

Projektplanering

Torsdag V.3	Planbeskrivning
Måndag V.4	Skriva mer i planen

Torsdag V.4	Sätta mig ner med Sigge och planera ut vad vi exakt ska göra varje dag.
Måndag V.5	programmera
Torsdag V.5	Programmera
Måndag V.6	Programmera
Torsdag V.6	Programmera
Måndag V.7	Programmera
Torsdag V.7	Förberedning inför måndag.
Måndag V.8	Redovisning och inlämning
Torsdag V.8	Övriga redovisningar

Kommenterad [A3]: måste göras denna veckan! V.3

Kommenterad [A4]: Sätt delmål för vad ni ska fokusera på. Programering tar tid men längs med vägen finns det ändå delmål tänker jag mig. Ingen avsatt tid för troubleshooting och bugfix exempelvis?

Ska det finnas UI?

Hur testas funktionaliteten?

Vad måste du lära dig mer om?

Jag måste lära mig mer om hur man enkelt kopplar ihop flera sockets utan att tappa kopplingen till de gamla. Detta är nog inte ett så stort problem och borde vara ganska enkelt att lära sig.

Har du andra tankar eller funderingar kring ditt projekt?

Jag kommer behöva spenderar ett bra tag för att programmera så det kommer vara de stora fokuset under projektets gång. Sen kommer vi även behöva testa rat en och se om den funkar på en global skala.

Lektionsutvärdering: Berätta hur det gått för dig idag. Jag tycker att det har gått bra jag har läst ganska mycket om hur socket nivån funkar och jag vet ganska klart hur jag kommer använda den för att bygga projektet.

[redacted]