Kontaktuppgifter

## Flutter: [achristoffer@ymail.com](mailto:achristoffer@ymail.com) React: aziz.bisevac@hotmail.com Angular: pontusthompsons@gmail.com Xamarin: [johanna.tjus@hotmail.com](mailto:johanna.tjus@hotmail.com)

**Sammanfattning**

För de fyra valda ramverken: Flutter, React, Angular och Xamarin, sammanställdes poäng inom olika områden. Sedan summeras dessa poäng, för respektive ramverk, för att presentera totalbetyget, som syns allra längst ner i matrisen nedan. Totalbetyget för varje ramverk görs upp genom adderingen av samtliga 1-5 poäng i alla 7 olika områden.

Rekommendering av programvara vilar på totalbetyget. Av denna slutsats, går det att poängtera att Flutter med hela 33 poäng, tar första plats följt av React med 30,5 poäng.

**Motivation och Rekommendation**

Vi valde att välja bort både Angular och Xamarin då de inte uppfyllde kraven lika bra som React och Flutter (se matris nedan). Detta innebar att vi gick vidare med React och Flutter och diskuterade kring vilket som skulle passa uppdragsgivaren bäst av dessa två.

Båda ramverken är väldigt kraftfulla och effektiva. Båda ramverken uppfyller kraven med höga poäng, så för att besluta vilket som kommer att vara bäst för framtiden har vi lagt större vikt på bakomliggande utveckling och stöd och utifrån detta kommit fram till en rekommendation.

Med utgång från det har vi då valt att rekommendera Flutter.  
Flutter är ett nyare ramverk och har väldigt stor potential och användar-skaran kommer att öka. Det finns mycket stöd och utbildningsmaterial. Då det är Google som tillhandahåller Flutter kommer det finnas mycket stöd i fortsatt framtid också.

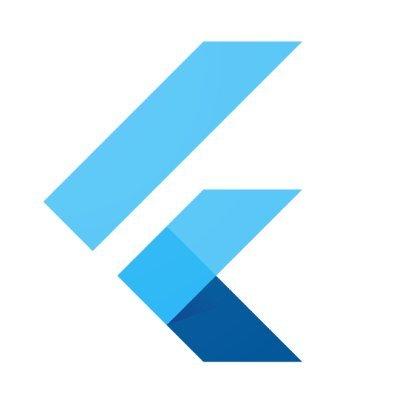
**Matris**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Krav** | **React: Väl uppfyllt krav(1-5)** | **Flutter:**  **Väl uppfyllt krav(1-5)** | **Angular:**  **Väl uppfyllt krav(1-5)** | **Xamarin:**  **Väl uppfyllt krav(1-5)** |
| **Stöd för att integrera kamera i applikationen** | **5** | **5** | **4** | **3** |
| **Stöd för både Android och iOS** | **5** | **5** | **5** | **4** |
| **Kunna hantera Web-api/Json** | **5** | **5** | **5** | **4** |
| **Inga licenskostnader** | **5** | **5** | **5** | **2** |
| **Vara populärt och kommande, dvs att den används i många tillämpningar och att använda skaran växer.** | **3,5** | **4** | **3** | **3,5** |
| **Snabbt och kompakt** | **4** | **5** | **3** | **3** |
| **Känsla under användning** | **3** | **4** | **2.5** | **2** |
| **Total** | **30.5** | **33** | **25** | **21.5** |

**Dokumentation kring ramverken**

Vi har tagit våra dokument där vi fördjupade oss kring de olika ramverken och lagt in dessa nedan så att all information är samlat i ett och samma dokument. Bifogar också länk till Github där dokumenten ligger separat och är kod också finns.   
  
Länk till Github: <https://github.com/HouseLife/Grupp7>

**Flutter**



- Används för att skapa native appar för både iOS och Android

- Tillgänglighet till API:er som kamera eller mikrofonen

- Det behövs bara en kodbas (dart) för att skapa appen vare sig för iOS eller Android

**Fördelar**

- En kodbas

o att ingen separat kod för iOS eller Android behövs.

o Detta innebära att applikationen ej behöver uppdateras 2 gånger.

- Layout metoder är lånade från den responsiva-webben (ungefär som färdiga bootstrap mallar)

- Ramverket skapar per automatik färdiga designmallar, ifall användaren ej vill själv designa applikationen.

- Mycket bra support och användarmanualer på Flutter.com

- Hot reload ger en snabbare insikt i dina förändringar i koden

- Nytt ramverk med stigande popularitet

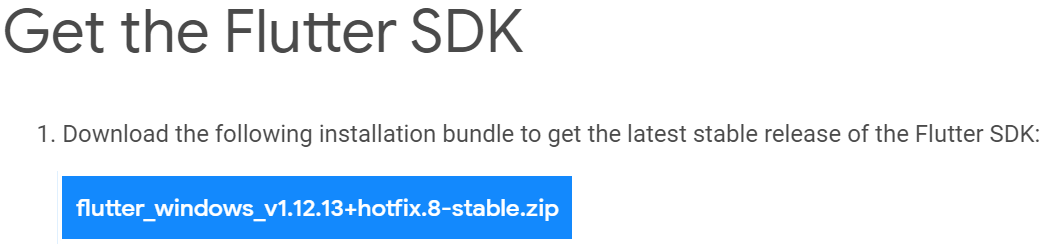
- Använder objekt orienterade programmeringsspråket Dart, enkelt att komma igång med ifall man tidigare arbetat med C# eller Java.

**Installera och lägga upp Flutter för Windows**

* **Steg 1: Android studio** (rekommenderas) eller **Visual Studio Code** (för lätthet)  
  + Android studio:
  + Det går att installera tillägg som hjälper till vid skapandet av applikationer med Flutter.
  + För att ladda ner: [www.developer.android.com/studio](http://www.developer.android.com/studio)
* Visual Studio Code (**används i detta fall**):
  + - För att ladda ner: <https://code.visualstudio.com/>

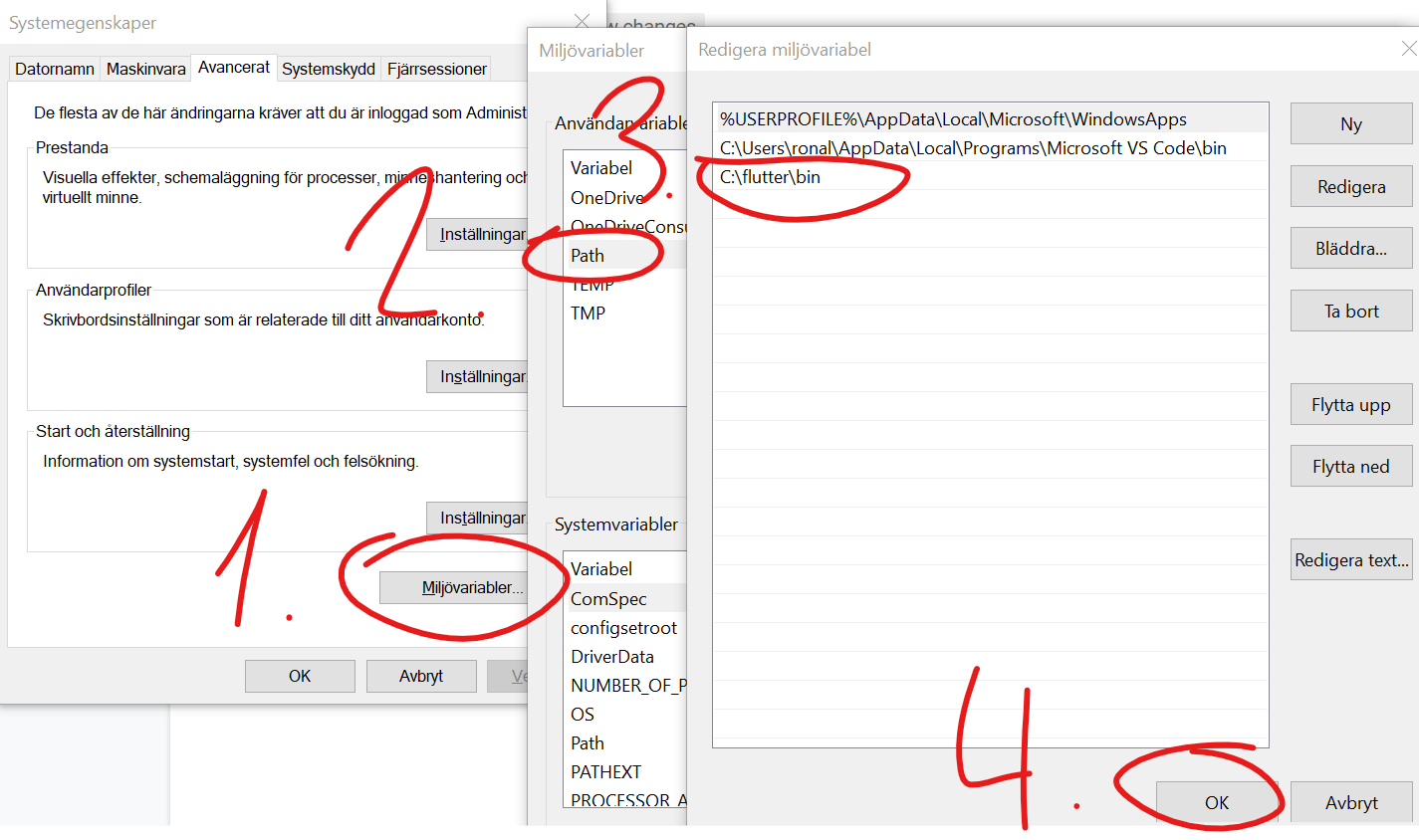
- **Steg 2: Installera Flutter**

* 2.1 För att ladda ner och installera, tryck på följande länk: [www.flutter.dev/docs/get-started/install](http://www.flutter.dev/docs/get-started/install)
  + För Windows, trycker du på Windows-knappen som presenteras hos flutter.dev, vilket redigerar dig till sidan “Windows install”
  + Scrolla ner till “Get the Flutter SDK” och under där trycker du på den blåa knappen som syns på illustrationen under.

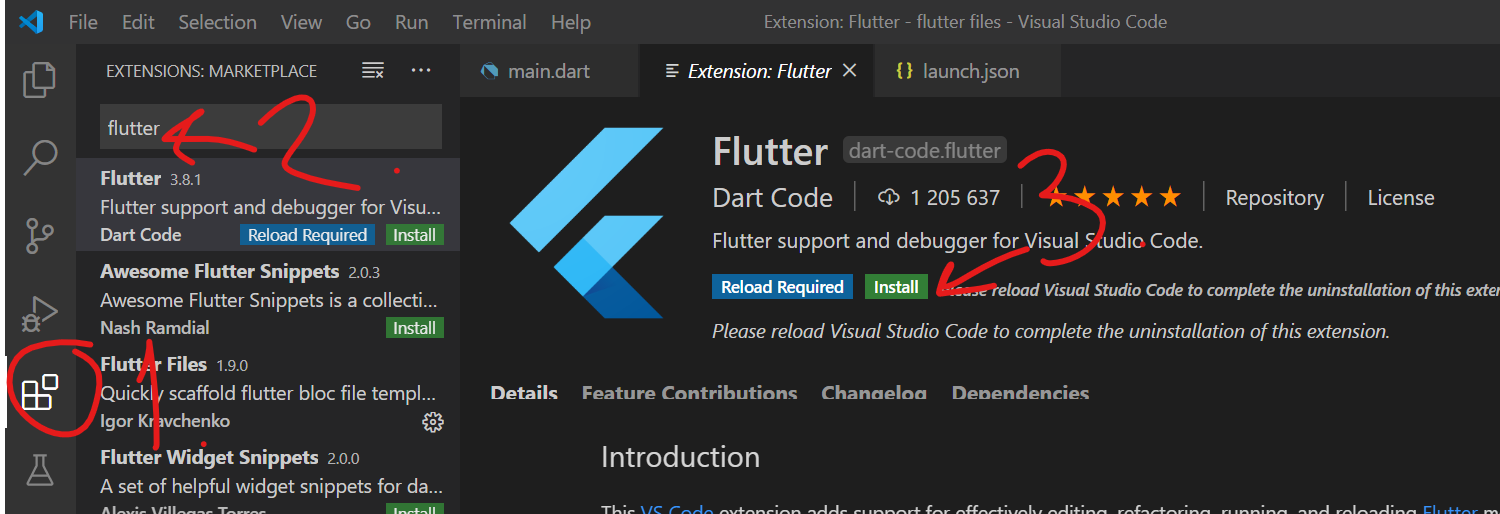


* 2.2 Då “flutter\_windows-v1.12.12+hotfix.8\_stable-zip” är en zip fil, måste den först packas upp.
* 2.3 Väl uppackat i Windows Utforskaren, skall denna mapp kallad “flutter” med all essentiell data, kopieras över till rotnivån på “C:\”-hårddisken, för att undkomma behörighets-lås som förhindrar flutters installering.
* Nu när Flutter är installerat, kan du endast köra det via ”Flutter Console” som är lokaliserad i den nya mappen med Flutters innehåll, “flutter” som kopierades in till “C:\”.
* Command Prompt för Flutter i Flutter Console:  
  + För att kolla diagnostik för om det är korrekt installerat eller inte, eller om du har programvaror som är kompatibla med Flutter eller inte, skriv där in följande:  
    - ”C:\Users\Shaun>**flutter doctor**”
* 2.4 \*OBLIGATORISKT\* - Om du vill kunna köra flutter på andra ställen i Windows (som i detta fall i Visual Studio Code, måste Path variabeln ändras i ”Redigera systemets miljövariabler”. Detta fönster kan sökas upp från Windows ”Start”.

* 2.4.1 I ”Redigera systemets miljövariabler”, tryck på ”Miljövariabler” och dubbelklicka sedan på filvägen i listan med ”Path” i namnet.



* 2.4.2 I detta fall måste ”C:\flutter\bin” finnas (eller skapas) om Flutter skall kunna användas från någon annanstans i operativsysystemet (i exempelvis Command Prompt eller Visual Studio Code).
* 2.5 \*Extra\* Nu kan du i Windows Command Prompt göra ett Command Prompt för Flutter och kolla status på vilken version som är installerad.   
  + 2.5.1 För att visa upp versionen som är installerad, skall det matas in följande i Windows Command Prompt: ”C:\Users\Shaun>**flutter --version**”
* **Steg 3: Installera Visual Studio Code samt tillägg för Flutter**  
  + 3.1 Ladda ner och installera Visual Studio Code.   
    - 3.1.1 du kan ladda ner Visual Studio Code härifrån: <https://code.visualstudio.com/>
  + 3.2 Väl installerat, starta Visual Studio Code, och ladda ner några tillägg för flutter miljön (illustreras i bilden nedan). För att göra det:   
    - 3.2.1 Tryck på knappen “Extensions” i vänstra hörnet
    - 3.2.2 Mata sedan in i inmatningsfältet “flutter” för att få fram resultat för tillägget “Flutter”, och tryck upp det.
    - 3.2.3 tryck på “Install” för att installera tillägget. Detta installerar vidare även ännu ett tillägg per automatik, kallat “Dart Code”.



* **Steg 4: Skapa en flutter applikation**
* Börja med att skapa en mapp i antingen, där app-projekt skall sparas, i utforskaren, någonstans på hårddisken. Vi kallar mappen “flutter files”. Sedan drar du bara filmappen och klistrar in den i Visual Studio Code, i fliken “Explorer”.
* För att skapa en ny app skall man nu trycka på fliken “Terminal”, sedan “New terminal”. Detta öppnar då upp ett console-fönster längst ner i Visual Studio Code.
  + För att skapa ett nytt projekt skriver du in i console-fönstrett: “flutter create projekt\_namn” och sedan trycker du “Enter” och applikationen skapas och läggs upp av programmet.

**Youtube-klipp:**

* “What is Flutter?”

<https://www.youtube.com/watch?v=I9ceqw5Ny-4>

* “Setup Flutter and Visual Studio Code In Windows”

<https://www.youtube.com/watch?v=5izFFbdHnWY&t=296s>

**Andra länkar:**

* Kortfattat Flutter vs React Native vs Xamarin:

<https://www.codementor.io/@mahil/react-native-vs-flutter-vs-xamarin-a-comparative-guide-2020-11evx1f7q4>

* Wikipedia (med Hello World exempel):

<https://en.wikipedia.org/wiki/Flutter_(software)>

* Mac

<https://medium.com/flutter-community/flutter-and-the-command-line-a-love-story-a3648ef2411>

****

React Native är ett ramverk som är tillverkad och underhålls av facebook. React Native har öppen källkod och släpptes 2015. Med React Native finns det möjlighet att skapa nativa applikationer för både till Android och Ios plattform. Detta simplifierar utveckling av applikationer till olika plattform då man inte behöver skapa separata kod. När man skapa applikationer för Ios och Android finns det möjlighet att exportera koder till nativa miljö så att t.ex Android studio kan förstå den exporterade koder. React Native använder sig av JavaScript och XML som programmeringsspråk. JavaScript är relativt enkel programmeringsspråk som rekommenderas till nybörjare inom programmering. Javascript inte bara enkelt men är även och kraftfull då det inte bara kan användas att skapa webbsidor men även applikationer.

**Fördelar**

**Tidseffektivitet**

Med ”Hot reloading ” har man möjlighet att kunna se live ändringar man gör, tex om man skapar en komponent kan man se det direkt utan att behöva spara. Detta sparar tid att utveckla applikationer som du snabbt kan visualisera det du skapar.

**Prestanda**

Prestandan som kommer med react native ligger inte långt från nativa applikationer. Detta eftersom när man utvecklar applikation på react native måste språket som används översättas till det nativa språk till exempel Java för Android applikationer.

**Popularitet**

React Native har öppen källkod vilket betyder att vem som helst kan bidra till ramverken. Detta medför att ramverket byggs på med nya funktioner som gör ramverket ännu bättre. Med över 80 000 stjärnor på Github är det kvitto på att React Native är en av dom populäraste ramverken. Detta blir en fördel om man skulle fastna eller få problem så finns det goda möjligheter att få hjälp.

**Färdiga komponenter**

Med färdiga komponenter blir utveckling av applikationer mycket enklare och snabbare än om man skulle koda från början. Komponenter som används på React native är ryggraden på ramverket det är det som möjliggör utveckling av applikation effektiv.

**Sparar pengar**

React native sparar processtiden det tar att utveckla applikationer vilket i sin tur sparar pengar. Genom att använda samma koder till olika plattform så kan man undvika att anställa fler utvecklare. I vanliga fall när man vill skapar applikation till Android och Ios så behövs således två utvecklingsgrupper. Men med React native eliminerar man behovet av flera utvecklare för att skapa cross platform applikationer.

**Pålitliga och stabila applikationer**

React native applikationer är uppbyggda av komponenter vilket underlättar uppgradering och modifikationer av en applikation. Ibland kan det vara svårt att skilja en React native applikation och en native applikation. React native applikationer är i par med nativa applikationer i form av prestanda, detta skapar en bild av stabilitet för React native applikationer. För den som inte är familjär av applikation utveckling kommer att ha svårt att se skillnad på React native applikation och native applikation.

**Nackdelar**

**Liten samling av komponenter**

React native har relativt liten samling av färdiga komponenter för tillfället, men detta kommer troligtvis att öka i framtiden.

**Native kod krävs**

I vissa fall så krävs det nativa kod för att få nativa funktioner att fungera så som gps ,kamera etc . Detta kan medföra anställning av utliggare personal som förstår nativa koder så som C eller Java.

**App-prestanda är inte lika optimerad**

I jämförelse med ett rent native ramverk så är inte React native optimerad för avancerad funktionalitet såsom komplexa applikationer, men om man tillverkar enklare applikationer så blir det knappt skillnad.

**Brant inlärningskurva**

Om man inte förstår Html, Css och Javascript blir inlärningskurva svår. Man ska helst bemästra Javascript, html, css innan man börjar med React Native.

**Säkerheten är låg**

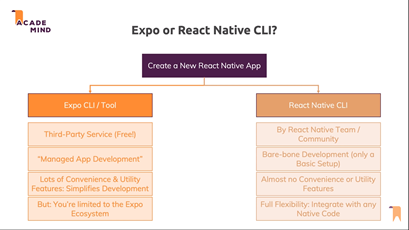
Eftersom React native är JavaScript-baserat bibliotek så är den sårbar mot attacker. Det gäller att använda koder som inte ger tillgång till säkerhetshål.

**Slutsats**

React native är bra för snabba och simpla applikationer. Tillverkar man stora och komplexa applikationer så får man vänta sig att prestanda tar stryk.

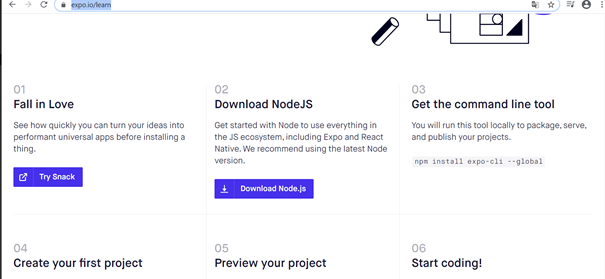
**Hello World Applikation**

Här nedan följer instruktioner hur man gör en Hello World Applikation på React Native.

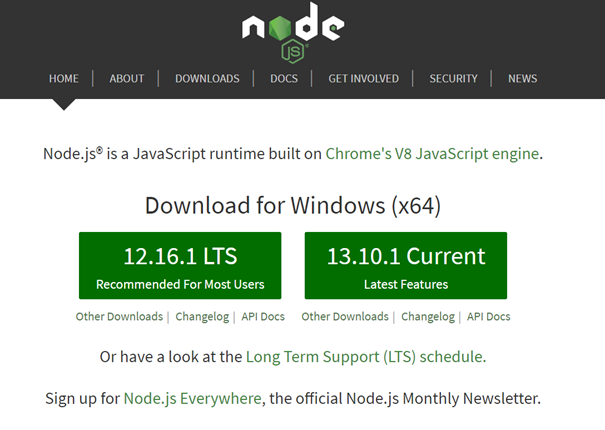


**Figur 1**

Innan man kan installera React Native så måste man bestämma sig vilket verktyg man vill använda sig av för att bygga applikationer. Det finns två alternativ Expo CLI och React Native CLI , skillnad mellan dom är Expo är lite lättare att komma igång och använda jämförelsevis React Native CLI som är mer för avancerade användare som ska bygga komplexa applikation . Det innebär att man möjligtvis behöver kunna nativa språk. Det bästa är att börja med Expo och eventuellt skifta till React Native CLI vid behov med hjälp av eject funktionen. För enkelhetens skull kommer vi använda oss av Expo för att göra Hello World app. Det första vi ska göra är att gå in på https://expo.io/learn . Här kan du börja med att klicka på Download Node.Js se figur 2.

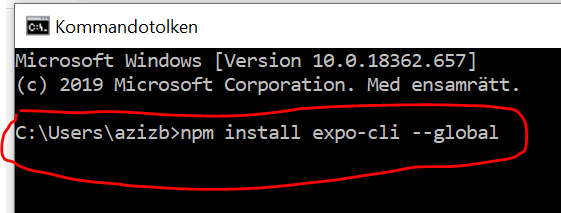


**Figur 2**



**Figur 3**

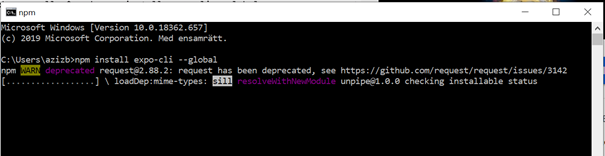
Här på figur 3 väljer du det senaste version av Node.Js



**Figur 4**

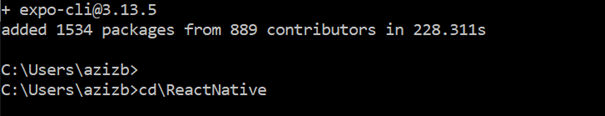
Efter installationen av NodeJs kan du gå in på kommandotolken och skriva in ” npm install expo-cli --global”.

Låt programmet köras kommer att se ut som figur 5 nedan.

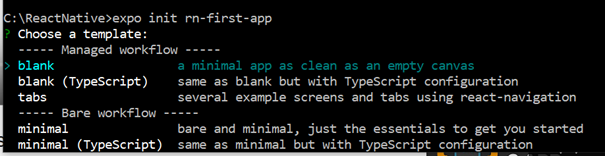


**Figur 5**

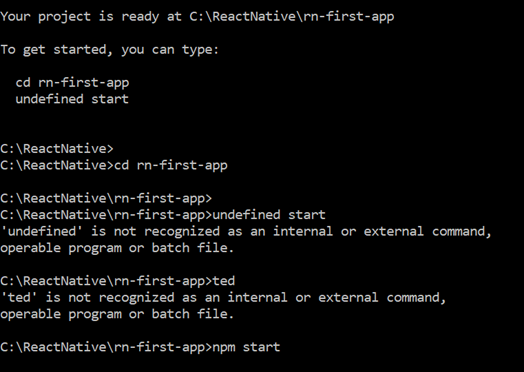
När installationen är färdig ska man skapa en mapp som skall användas till projektet detta gör du genom att skriva cd\Namnge Mapp här, i mitt fall skapade jag mapp som heter React Native på c:\ partition. Se figur 6 nedan.

**Figur 6**

**Figur 7**

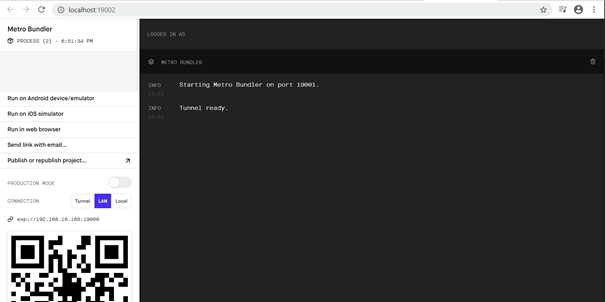


På figur 7 ska man välja template , vi kommer att välja ”Blank” för att detta räcker för HelloWorld app.



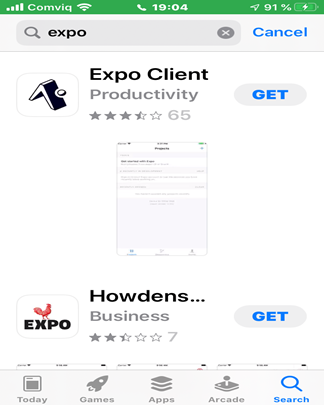
**Figur 8**

När man valt template kan man starta expo genom att skriva cd rn-first-app sedan tryck på enter. Sedan skriver man npm start för att starta Expo.



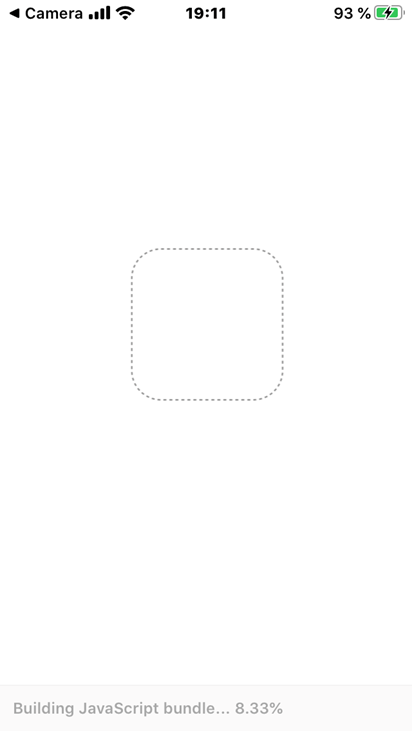
**Figur 9**

När man skriver npm start så får man upp denna tab på webbläsaren automatiskt. Här kan man välja vilken emulator man vill använda, det finns Ios och Android emulator.



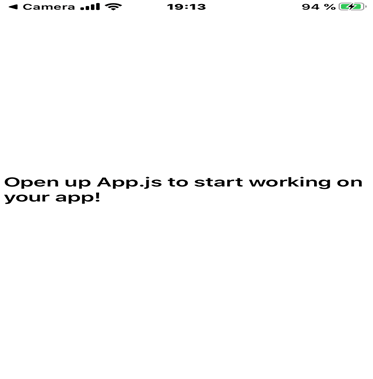
**Figur 10**

För att använda Expo på telefonen så kan man ladda ner Expo på både Android och Iphone. Här på figur 10 illustreras hur man laddar ner Expo på Iphone. Detta gör man man genom att gå in på appstore och söka ”Expo” Välj sedan ”Expo client”.



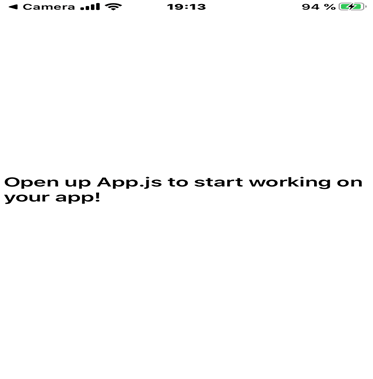
**Figur 11**

När man installerad applikationen skall man scanna QR-koden som finns på expo sida på figur 9. För mig funkade inte tills jag valde ”Tunnel” som connection.



**Figur 12**

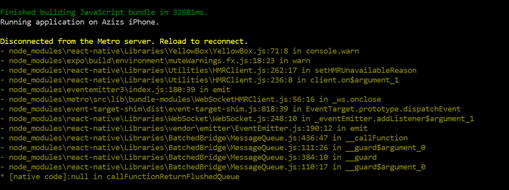
När applikation är klar för att använda får man denna meddelandet på figur 12. Detta betyder att kopplingen är klar och det är då dags att koda. Jag kommer att använda mig av visual studio code för att programmera vilket är rekommenderat och dom flesta använder Visual studio code.



**Figur 13**

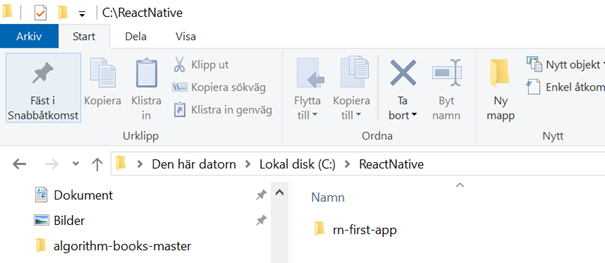
Här visas bild på kommandotolken när telefonen är kopplad. Man ska inte stänga ner denna koppling eller kommando tolken för att kopplingen till telefonen ska ha funktionen Hot reload. Så att man ser i visuellt vad som händer medans man kodar.

**Figur 14**

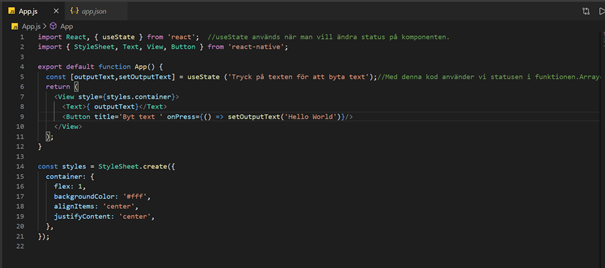


Här illustreras visual studio code den applikationen vi skapade med kommandotolken genom att öppna mappen vi skapade se figur 14.

**Figur 15**



När vi öppnar visual studio code och väljer rotmappen rn-first-app så får vi upp koderna på figur 13. Vi ska göra en enkel applikation. När man trycker på en knapp eller text så ska texten bytas till Hello World.

**Figur 16**

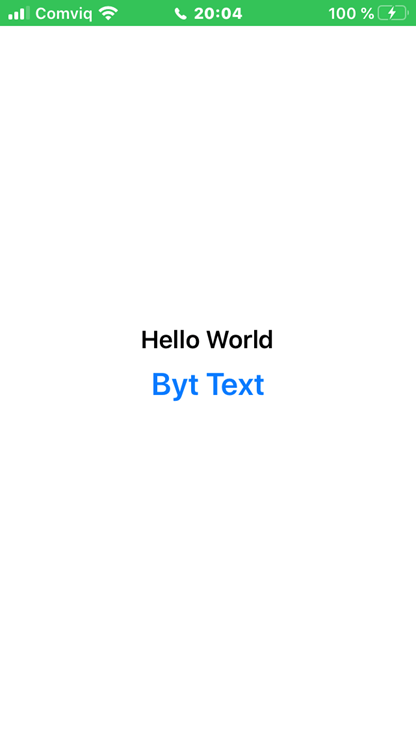
Här illustreras bild efter vi har kodat vår första applikation. Applikationen är enkel, när man trycker på texten på ”Byt text” då kommer texten ”Hello World” visas på telefonen. Se bild nedan.

**Figur 17**



Här illustreras applikationen Hello world, genom att klicka på knappen ”Byt text” får man upp Hello World se bild nedan.

**Figur 18**



På figur 17 illustreras utskrift av Hello world på Iphone.

**Reflektion**

Vi tyckte att det var relativt enkelt från att installera alla nödvändiga program till att göra en Hello world Applikation. Det förutsätter att man använder Expo CLI då det är lättare att komma igång. Om man ska göra lite mer avancerad applikation då väljer man React Native CLI men då gäller det att man kan nativa språket till respektive plattform. Vi valde Expo CLI som verktyg för att det är mer nybörjarvänlig. Men rekommendationen är att alltid börja med användning av Expo och sen gå över till Native Cli om det finns behov för det.

**Källa:**

<https://www.youtube.com/watch?v=qSRrxpdMpVc> : följde denna video för Hello World appen.<https://expo.io/learn> : Använde denna sida för att installera alla nödvändiga filer.

<https://citrusbits.com/react-native-pros-and-cons/>

**React start**

(Hello world)

Detta är en beskrivning på hur man gör en Hello world på vanliga React och inte React Native.

Först behövs det att ladda ner två program. den första är Node.js och den andra är en

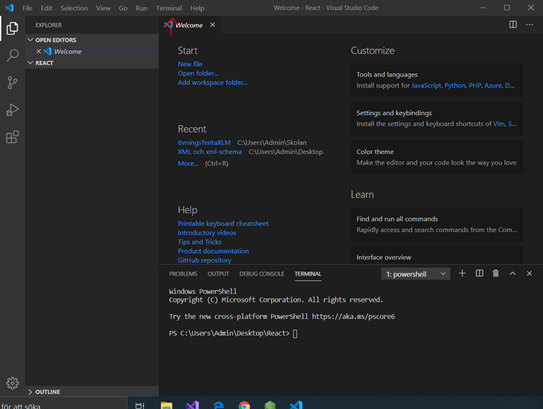
textredigerare av ditt val. vi har valt att använda visual code som textredigerare. Det Node gör för dig är att den är en ganska bra plug in för att skapa olika applikationer med deras stora bibliotek med olika koder.

Vi har använt oss av en youtube tutorial för att skapa denna Hello world, länken till videon ligger längst ner. Det finns 100 tals olika tutorials på hur man börjar använda React så om inte detta hjälper dig att skapa en hello world så kan en annan tutorial kanske hjälpa.

Create React-app är en bekväm miljö för att lära sig React. Det sätter upp din utvecklingsmiljö så att du kan använda de senaste JavaScript-funktionerna. Den gör så att man behöver bara skriva en order och den skapar en hel react projekt åt dig.

steg 1.

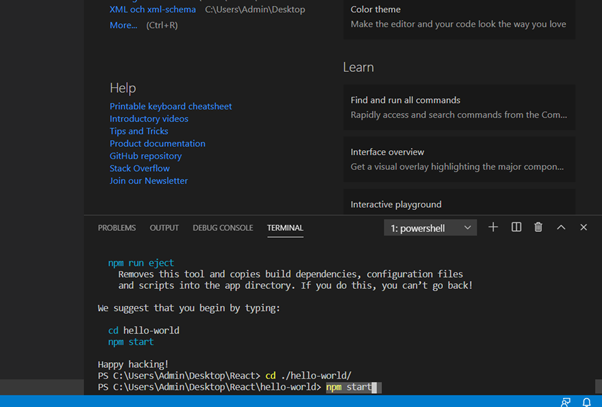
öppna visual code och välj en mapp. Efter det så ska du klicka på terminal och välja en new terminal.



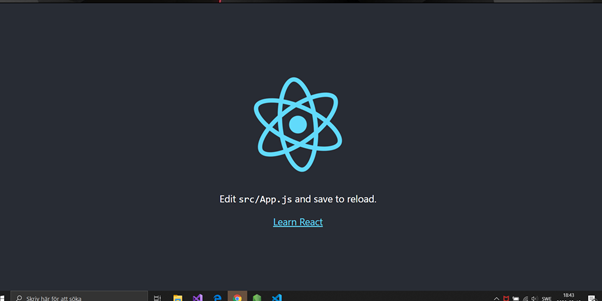
Efter du har klickat på new terminal så poppar det upp en liten sida längst ner på visual code och där ska du skriva npx create-react-app och namn på appen du vill skapa, för just nu så kallar vi appen hello world. Man kan inte använda stora bokstäver för namnet på appen. När nedladdningen så har du en ny mapp som heter hello world på vänstra sidan.

steg 2.

För att köra denna applikationen så behöver man navigera först i mappen. man behöver skriva in längst ner på samma fönster cd ./hello-world/ och klickar enter, under den så behövs det npm start för att starta.



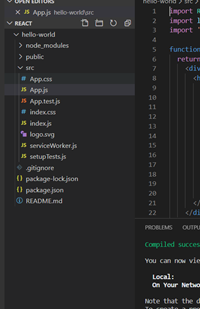
efter att man har klickat enter så tar den dig till en sida i google eller den du har valt att använda.



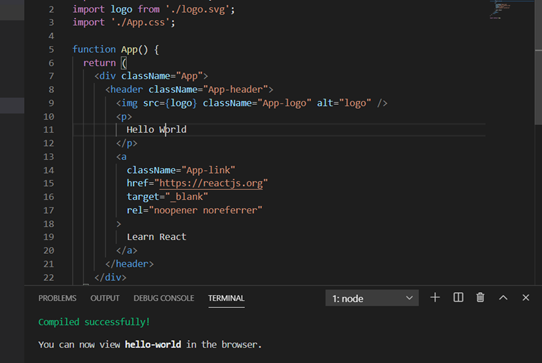
den ger dig instruktioner på hur du ska starta.

Steg 3.

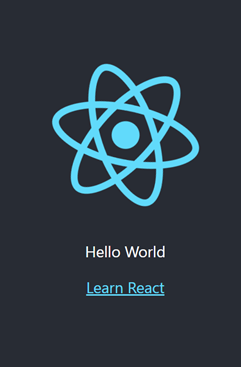
Gå tillbaka till visual code och öppna mappen hello-world, efter det öppnar du src och väljer App.js



För att ändra text så gör man det mellan <p> och </p>. vi kan börja med att skriva in Hello world där.



om jag sparar filen nu så ändras det direkt på sidan utan att man behöver starta om den och det är en funktion som react har.



Det är verkligen så lätt att börja skapa en app, hoppas detta har varit lärorikt!.

Youtube tutorial :<https://www.youtube.com/watch?v=9hb_0TZ_MVI>

**Angular**



**Information**

Angular är ett komponent styrt ramverk. Angular är även konstruerat för att bygga *client-side* applikationer, vilket innebär att aktiviteter utförs av en klienten i en klient server relation. Applikationer som använder sig mycket av javascript är passande för Angular.

Angular använder sig av ett tillvägagångssätt som betonar separationen av funktionaliteter av ett program, till oberoende och utbytbara moduler. Detta innebär att programmet är uppdelat i flera moduler med olika funktionaliteter och syften. Detta är bra då det möjliggör att programmerare kan dela upp arbete enklare då olika funktionaliteter uppdelas i egna moduler. Då Angular är komponent styrt medför det att det blir enkelt att återanvända kod.

Google har bestämt att uppdatera Angular två gånger om året där de använder sig av semantisk versionshantering. Semantisk versionshantering innebär att uppdateringen av ett program eller applikation är uppdelad i tre olika steg, som ser ut något liknande: version: 2.3.4. Den tredje siffran i detta exemplet “4” ökar när buggfixar eller andra mindre justeringar utförs. Den andra siffran “3” ökar ifall nya funktioner adderas till applikationen utan att ändra redan existerande funktionaliteter. Den första siffran “2” ökar ifall funktionalitet förändras i applikationen. Då Google uppdaterar Angular två gånger om året så ökar den första siffran två gånger om året.

**Fördelar**

* Angular avlämnar bekvämt utvecklaren att aktivt manipulera DOM (Document Object Model)
* Vinkelformat minskar belastningen från server-CPU: er. Det betyder att servern presterar bra tack vare minskad trafik och eftersom den bara serverar statiska filer och svarar på API-samtal.
* Angular bygger mycket testbara webbplatser och applikationer, detta genom att tillåta enhetstestning och slutändningstester.
* Angular hanterar planeringen av resrutt vilket gör det enkelt att byta view.
* Angular erbjuder hög prestanda.
* I Angular kan man debugga mallar.
* Angular är modulbaserat vilket innebär att koden är separerad och organiserad.

**Nackdelar**

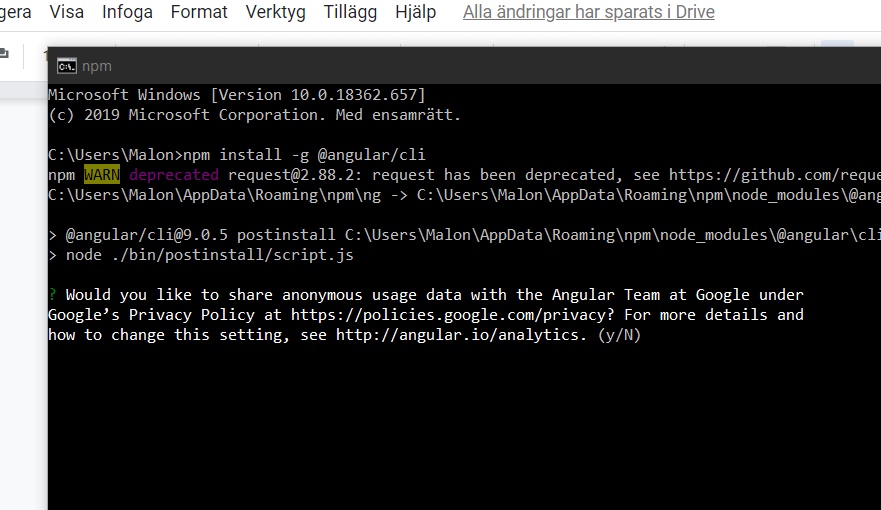
* Ganska skiktade och arrangerade hierarkiskt kan omfång vara komplexa enheter att hantera om det är din första upplevelse med Angular
* Istället för att följa en rak inlärnings graf kan du behöva möta stora svårigheter att anpassa dig till ramverket.
* Angular funktionaliteter kan vara svåra för nybörjare att förstå vilket innebär att inlärningskurvan är stor.
* Angular uppdateras två gånger om året vilket ofta tvingar användare att anpassa sig.
* Tredje part integrationer kan vara komplicerade.

**Process att skapa hello world samt emulera hello world**

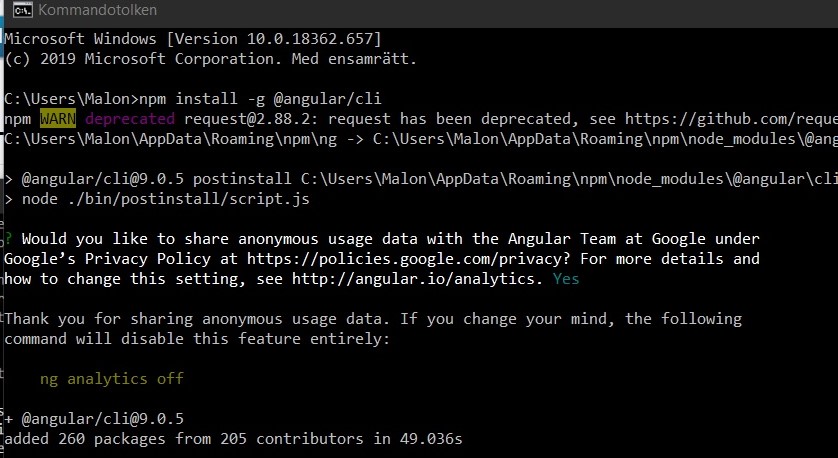
* Det första steget blev att söka efter en youtube video med sökningen “First app with angular”.
  + Vi fick upp flera bra videoklipp och även en serie med massor av videoklipp där en person går igenom både kod och information gällande angular.
  + Här är serien av youtubeklipp: [Angular 8 Tutorial](https://www.youtube.com/playlist?list=PLC3y8-rFHvwhBRAgFinJR8KHIrCdTkZcZ)
  + Vi använde oss mestadels av klipp 1 - 3.
    - Klipp 1 = Viktig information gällande Angular.
    - Klipp 2 = Går igenom vilka program som skall laddas ner för att kunna påbörja programmering.
    - Klipp 3 = Visar hur man skapar en egen hello world app med Angular
* Andra steget var att faktiskt installera programvaran ***Node NPM*** som behövdes, där installationsfilen laddades ner på: <https://nodejs.org/en/>

## Därefter gick vi in på kommandotolken (CMD) och skrev in ***“node -v”*** samt ***“npm -v”*** för att verifiera att ***Node NPM*** blivit installerat korrekt.

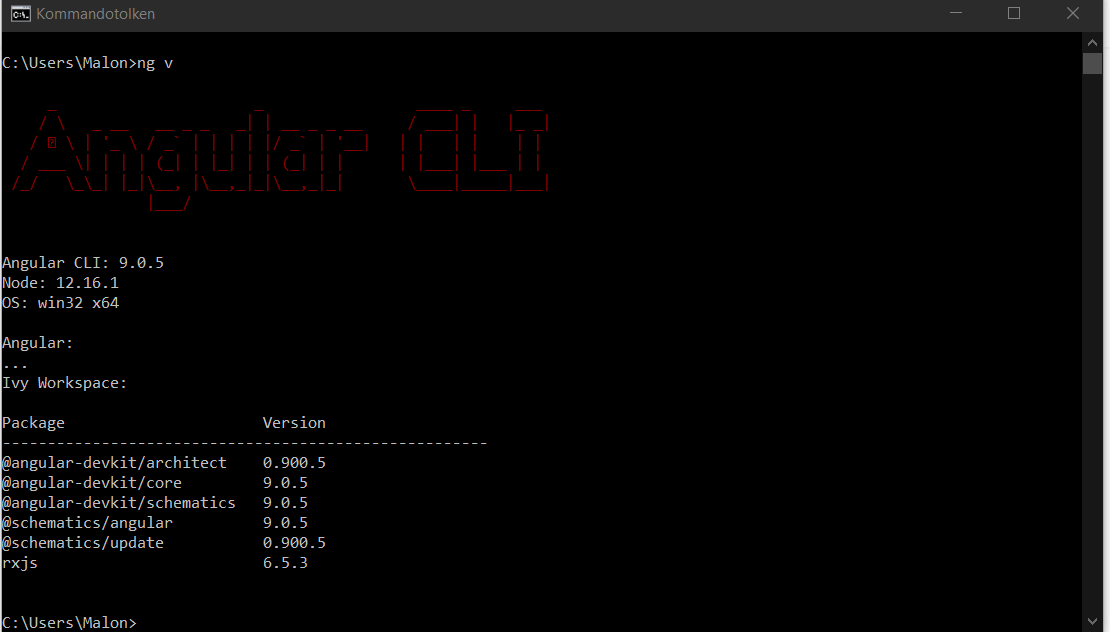
* + Härnäst går vi till: <http://cli.angular.io> där vi hämtar kommandorader som vi förde in i kommandotolken (CMD).
  + När vi skrev in ***“npm install -g @angular/cli”*** i kommandotolken (CMD) blev resultatet:



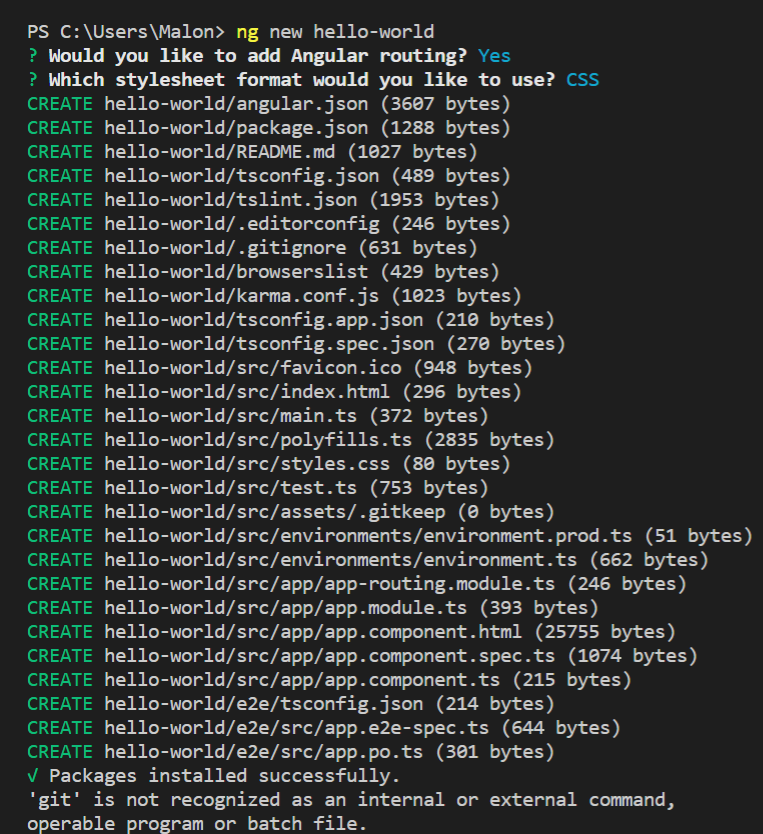
* + Vi accepterade att vår data skulle delas med Google genom att acceptera deras privacy policy och då skedde:

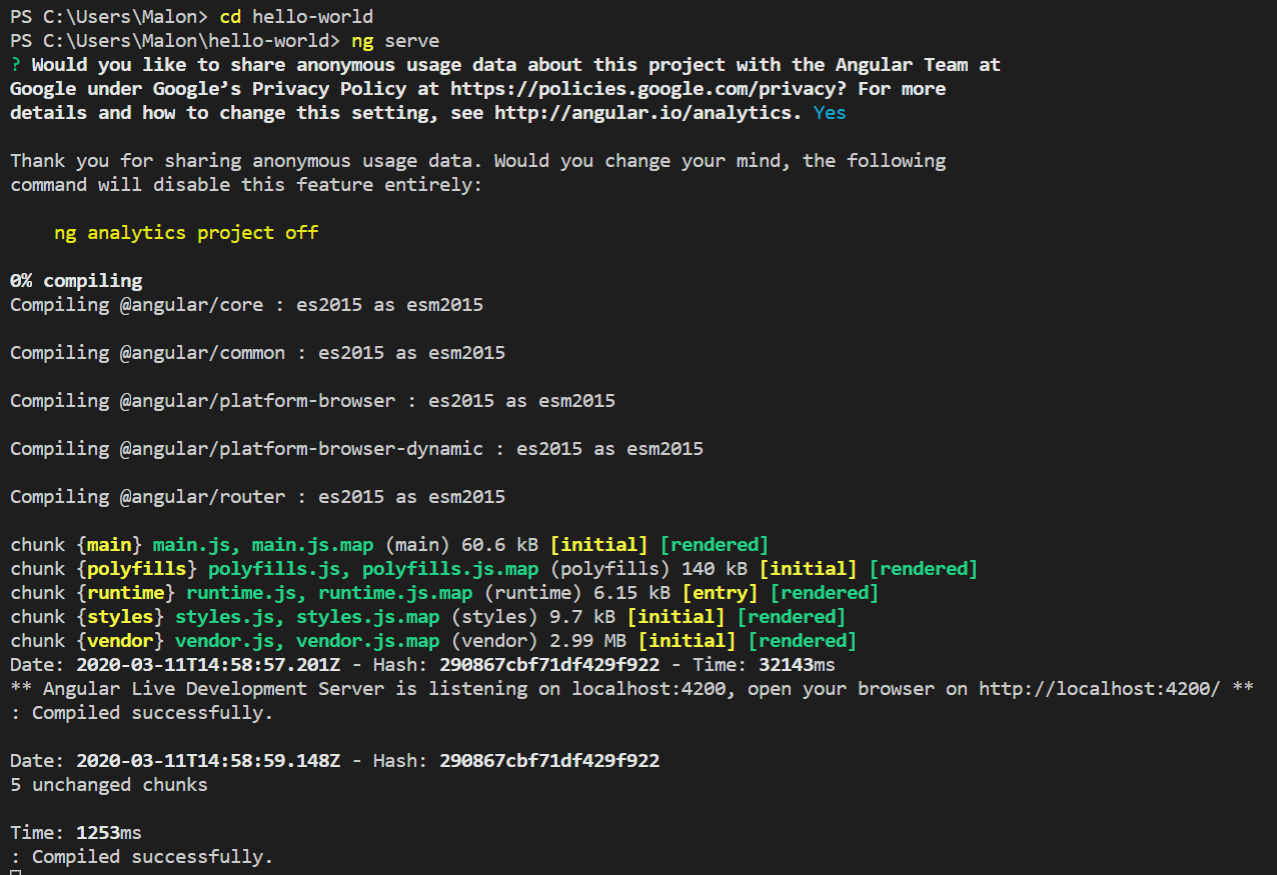


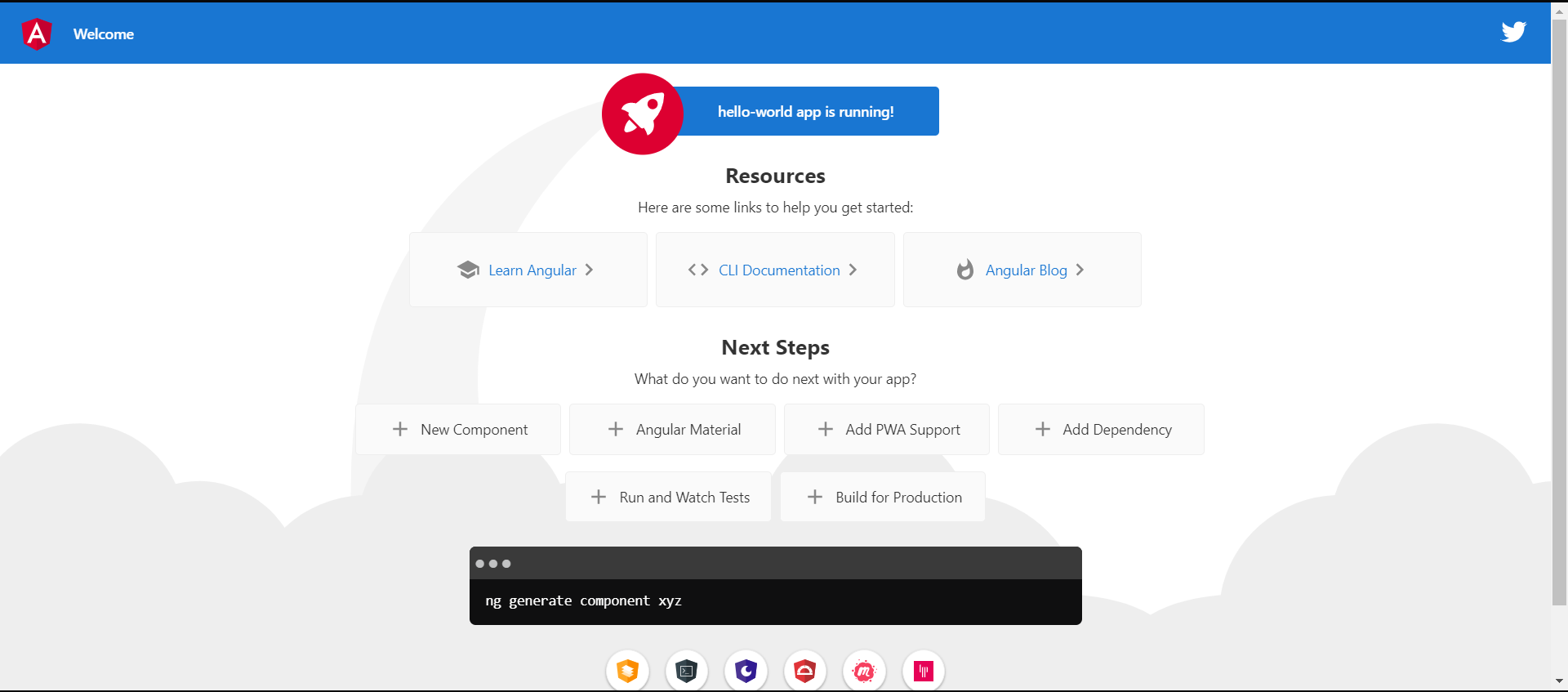
* + Härnäst kontrollerar vi ifall Angular CLI installerades korrekt genom att skriva in ***“ng v”*** i kommandotolken (CMD).

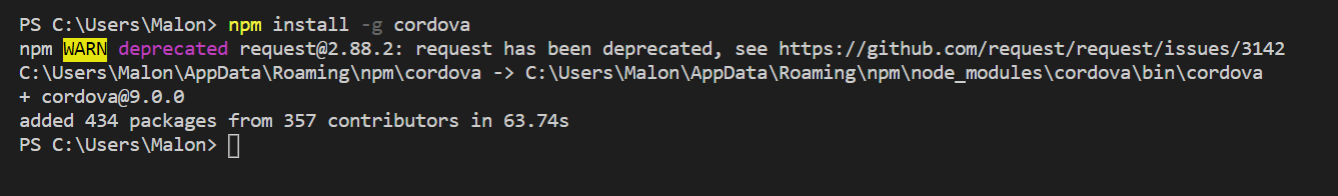


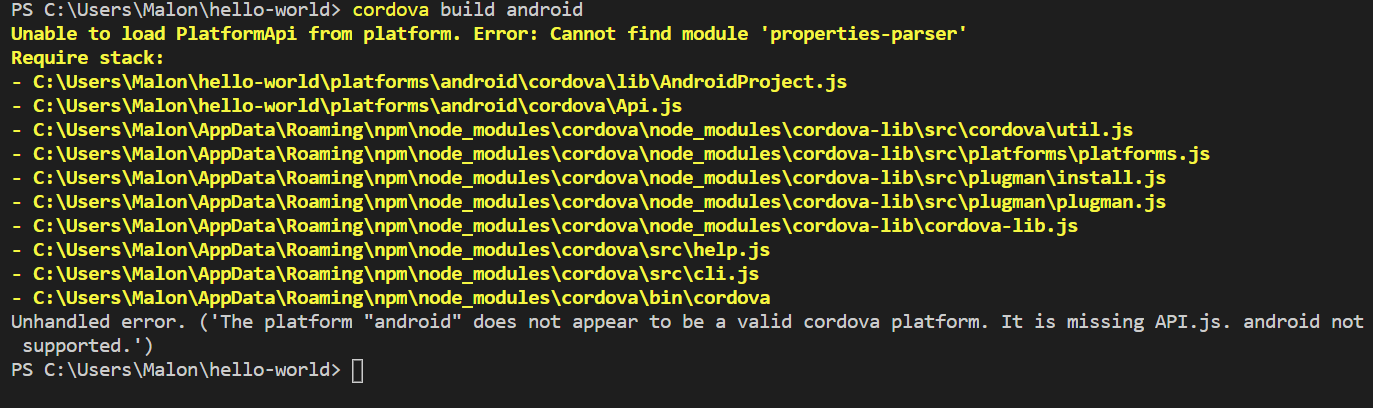
* Efter vi hade fört in raderna i kommandotolken (CMD) så öppnade vi visual studio code.Därefter öppnade vi mappen ***“Users\malon”*** där vi tidigare installerade Angular CLI.
  + Vi öppnade en integrerad terminal i mappen för att sedan skriva ***“ng new hell***
  + ***o-world”*** där hello world är projektets namn.



* + Efter installationen var klar lokalisera vi projektmappen som vi döpt hello-world genom att skriva ***“cd hello-world”*** i den interna terminalen.
  + Nu befinner vi oss i projektmappen och skrev därefter ***“ng serve”*** i terminalen för att starta applikationen.
  + Därefter skrev vi in ***“localhost:4200”*** i webbläsaren för att kontrollera att hello-world appen är igång.



* Nu vill vi få applikationen att köras i android och för och göra det måste vi ladda ner Cordova globalt.
  + Detta gör vi genom att skriva kommandot “npm install -g cordova” i terminal på visual studio code.
  + Hädanefter skapade vi ett Cordova projekt genom att skriva in ***“cordova create hello-app-world-cordova com.example.hello OurFirstApp”*** i terminalen.
  + Sedan ville vi navigera till den nyligen skapade cordova projektmappen (***hello-app-world-cordova***). I den mappen ville vi navigera till “***Platforms***” som vi gjorde genom att ändra terminalen till ***“hello-app-world-cordova”*** genom att skriva detta i terminalen:***“cd hello-app-world-cordova/platforms”.***
  + Nu befinner vi oss i mappen ***“platforms”*** där vi skriver in ***“cordova platform add android”*** i terminalen så nödvändiga resurser installeras.
  + Efter detta flyttade vi över mapparna och filerna: ***hooks, platforms, plugins, www*** och ***config.xml*** från den nyligen skapade ***hello-app-world-cordova*** till vår tidigare ***Angular hello-world*** applikation
  + Hädanefter sammanfogade vi vår ***Angular hello-world applikation “package.json”*** med ***hello-app-world-cordova “package.json”*** filen***.***
  + Sedan ändrade vi ***“ <base href=”/”> ”*** till ***“ <base href=”./”> ”*** i ***index.html*** i Angular projektet.
  + Därefter öppnade vi ***angular.json*** och ändrade ***outputpath*** till ***“www”.***
  + Efter detta öppnade vi ***tsconfig.json*** och ändrade ***compilerOptions target*** till ***es5*** från ***es2015.***
  + Sedan öppnade vi terminalen igen och matade in värdena ***“ng build --prod --aot”.***
  + Nu vid det slutliga skedet där vi skulle bygga android applikationen och emulera den så fick vi error och kunde inte åtgärda dessa. Vi försökte mata in ***“cordova build android”*** och ***“cordova emulate android”*** i terminalen, men när vi matade in dessa så fick vi detta error meddelandet på båda inmatningarna:

Vi klarade inte av att emulera hello world applikationen på en mobil men fick hello world hemsidan att fungera.

**Åsikter**

Vi tycker att Angular verkar vara ett bra ramverk eftersom det finns en Community sen många år tillbaka som arbetat med det. Det finns massor av information och video redovisningar om hur man använder det och även personer som redovisar tips.

Trots att det är ett välanvänt ramverk vill vi fortfarande inte välja just detta ramverk på grund dess ålder samt att populariteten inte växer.

Det är absolut en värdig kandidat till resterande utvalda ramverk men när det finns nyare ramverk som växer i popularitet vilket är ett stort kriterium vi följde under analysen av ramverk. Därför anser vi trots Angulars styrkor att Flutter eller React är mer lämpligt för ditt projekt. Vi upplevde Angular som rörigt till en början då det är uppdelat men vi kan även se styrkor i detta ifall man har tidigare erfarenhet och lägger mer tid på ramverket. Vi upplevde transaktionen mellan hemsida till applikation som svår men detta kan ha påverkats av vår begränsade kunskap inom ramverk.



**Information om Xamarin**

* Xamarin använder språket C# och utvecklas med .NET. Man skriver all kod i C# och sedan kan man använda den koden för både IOS och Android.
* Tillgång till API
* Grattis och öppen källkod, inga license eller andra kostnader
* Xamarin erbjuder cross-platform support för android, iOS och Windows. Xamarin integreras också med tusentals tredjepartsbibliotek inklusive Google Play-tjänster, Facebook, Google API: er för iOS.
* Den kan dela upp till 90% av bas koden.
* Kombinerar allt från native plattformarna och inkluderar många verktyg själv
* Underhåll och uppdatering av appar byggda med Xamarin kräver mindre arbete. När du har gjort ändringar i källfilen kan de tillämpas direkt på dina appar, vilket eliminerar behovet av att uppdatera källkoden för dina appar individuellt om eventuella uppdateringar, buggfixar eller nya funktioner skulle bli nödvändiga.

Tutorial, vad är xamarin : <https://dotnet.microsoft.com/learn/xamarin>

**Process**

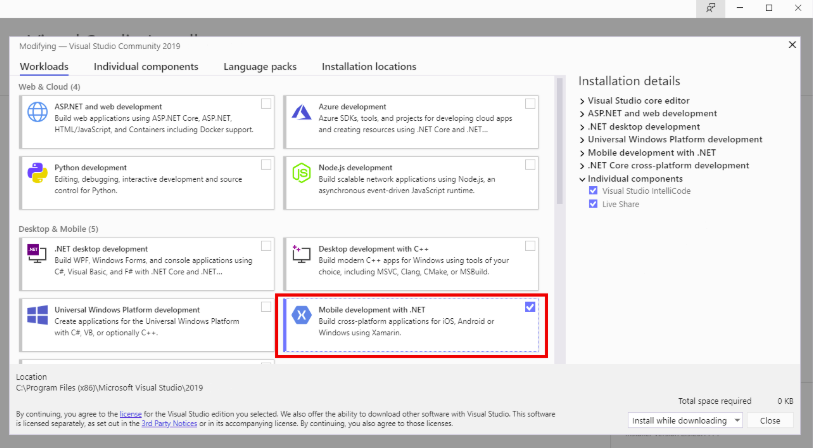
Nedladdning:[**https://dotnet.microsoft.com/apps/xamarin/mobile-apps#ios**](https://dotnet.microsoft.com/apps/xamarin/mobile-apps#ios)

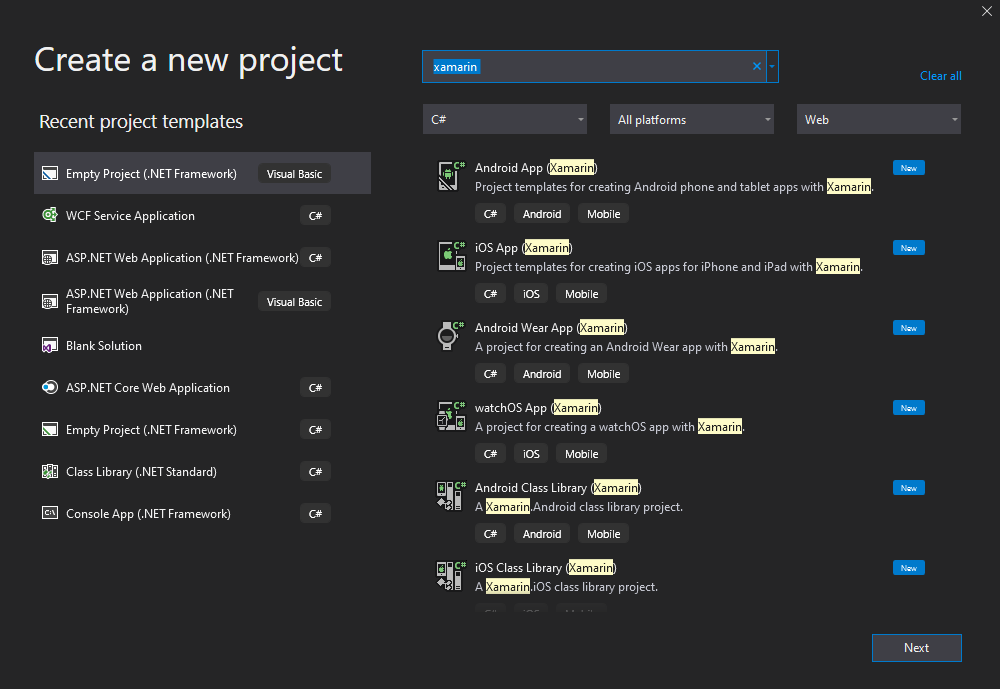
get started, Hello world in ten minutes:  
Om du följer denna tutorial steg för steg har du sedan en app för både iOS och Android gjord i Xamarin. Det finns beskrivning för både Microsoft och MAC.   
Vi har valt att inte skapa en egen process i detta dokument för hur man gör då denna beskrivning vi hittade är väldigt bra att använda, de har många länkar och bra beskrivningar.   
<https://dotnet.microsoft.com/learn/xamarin/hello-world-tutorial/intro>

**Visual studio:**

Börja med att ladda ner visual studio. länk(finns också i beskrivningen ovan) <https://visualstudio.microsoft.com/downloads/>

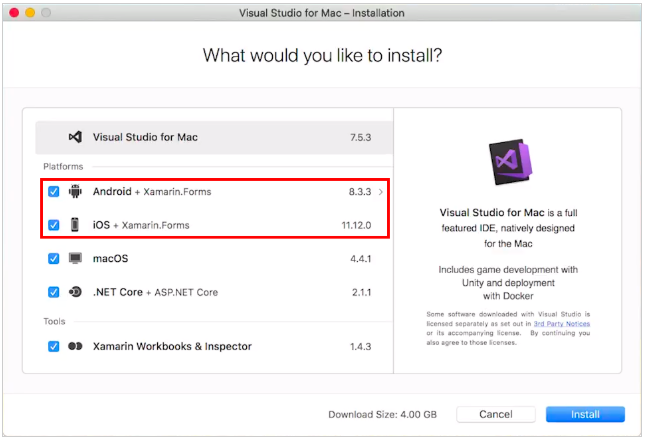
**Windows**:

När du laddar ner, välj: Mobile development with .NET. 



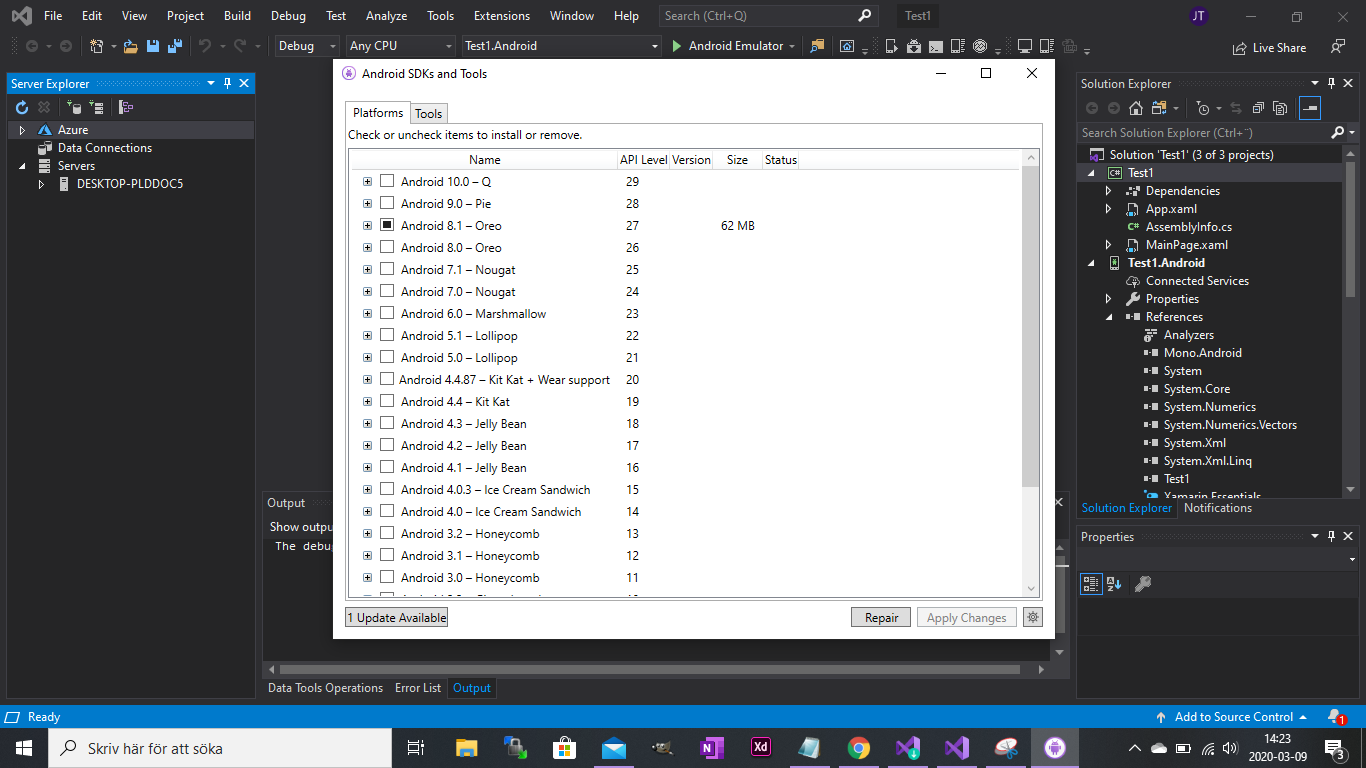
**MacOS:**

För macOS lägg till **Android + Xamarin.Forms** och  **iOS + Xamarin.Forms**



**Emulator**

<https://docs.microsoft.com/sv-se/xamarin/android/get-started/installation/android-sdk?tabs=windows>



Problem:

Vi stötte på problem när vi skulle försöka använda en emulator.

**Hur har det gått**

Det gick väldigt smidigt att lägga till Xamarin på visual studio och när man skapar projekt skapar den väldigt mycket åt en.

Det första problemet vi stötte på var att få igång Android simulatorn. Det tog väldigt mycket plats på datorn och det var svårt att hitta rätt, det var flera filer och inställningar som behövde utföras. Det tog väldigt lång tid.

Xamarin tar för mycket plats på de datorer vi har haft tillgång till därför kunde vi inte skapa en app i xamarin. Det kommer ta ännu mer plats när man lägger till plugins etc.

Det går att skriva all kod på en Windows dator, men man behöver koppla till en Mac för att skapa iOS applikationer, det fungerar inte att göra allt på en dator.

Därför har vi beslutat att Xamarin är ett dåligt val av ramverk.

**Kod i C# för “Hello world”**

kod i C# är väldigt lätt, så här ser det ut för “hello world”.

