React – Events

1. Skapa ett nytt projekt i react. Kommer du ihåg hur du skall göra? Det var någonting med npm create …
Se bara till att du står på ett bra ställe innan du börjar skriva kommandot så att ditt projekt hamnar på ett bra och strukturerat ställe. Helst inte i rooten någonstans.

2. Nu när du har skapat ett projekt behöver du gå in i mappen för projektet. Detta gör du fortfarande genom att i konsolen som du hade öppen när du skapade projektet skriva:
cd <namnet på projektet>
Då kommer du att hamna i projektmappen och därifrån kan du testa att ditt projekt fungerar. För att köra igång ett projekt, skriv
npm run dev
när du står i projektmappen.

3. Dags att börja skapa någonting. Men om det blev fel i förra uppgiften behöver du kanske skapa om projektet så att det går att köra ett projekt ordentligt innan du fortsätter.
Avbryt körningen av applikationen genom att trycka CTRL+C om du har en pc. Tror att det kan vara CMD+C på mac.
Öppna en editor (Visual Studio Code) genom att skriva
code .
i konsolen.

4. Nu vill jag att ni skall skapa en komponent. Kommer ni ihåg hur det går till? Skapa upp en mapp som heter components och däri kan ni skapa era filer. En komponent består av en fil med filändelsen tsx och eventuellt en fil med filändelsen css. Vad filen skall heta är upp till er, men jag vill att denna komponent skall fungera som en adderare, ni skall slå ihop två tal i slutändan.

5. I er nya komponent behöver ni skapa upp två stycken textrutor, en knapp och något

element för att visa resultatet. Detta göra inuti den return som ni har i er komponent.
Om ni vill ha en snabb repetition över hur en komponent är uppbyggd kommer det här:
Så ni behöver skriva html för de element som är listade ovan (i return). Om ni känner för lite styling går det jättebra att lägga till det också.

6. Om ni åter igen startar applikationen med npm start i konsolen kommer ni se att det inte syns någonting mer på skärmen än det gjorde sist. Det är tråkigt då vi faktiskt har skrivit en massa kod. Så öppna upp App.tsx, vilket är vår startpunkt, den komponent som alltid kommer att renderas när applikationen startar. I denna komponent vill jag att ni lägger till er nya komponent. Detta kommer göra att den syns i webbläsaren när applikationen startar. Här kan jag inte säga exakt vad ni skall skriva då er komponent heter lite olika från projekt till projekt.
<MyComponent></MyComponent>
Någonting som liknar det ovan behöver ni skriva någonstans i app-komponentens return.

7. Nu kommer ni att se allting ni har skrivit i er nya komponent, sweet! Dags att fortsätta.

8. Vi behöver nu ha möjlighet att fånga det som en användare skriver in i textrutorna. Detta kuna vi göra med hjälp av traditionell javascript genom att hitta id:t på vårt element och sedan plocka ut value från textrutan. Detta är lite gammeldags nu när vi har fina ramverk för våra applikationer. I denna repetition behöver vi använda oss av någonting som heter state för att hela tiden hålla informationen som användaren skriver i vår komponent.
Målet är att vi vill ha en variabel som hela tiden innehåller det som användaren har skrivit i sina textrutor.
Skapa en state-variabel genom att skriva följande:
const [firstNumber, setFirstNumber] = useState(0)
Detta betyder att ni kommer att ha en variabel som heter firstNumber som från början innehåller talet 0. Mycket bra!
För att ändra firstNumber MÅSTE ni använda funktionen som heter setFirstNumber(x). X är ett nytt tal som ni vill ändra firstNumber till.

9. Eftersom vi har två textrutor behöver vi två stycken state-variabler så passa på att kopiera den första och klistra in en till under men döp om den till secondNumber :)

10. Ok, så långt, toppen! Men, vi har än så länge bara två variabler som aldrig ändras och alltid kommer att innehålla talet 0. Vi behöver någon funktionalitet som gör att variablernas värde ändras. Detta gör vi enklast genom att skapa upp en change på våra textrutor. Det betyder att vi kan köra en funktion så fort någonting skrivs i våra textrutor. Niiice!
<input type='number' onChange={myAwesomeFunctionName} />

11. Nu behövs också funktionen som ni sade skulle köras i er change-händelse. Skapa upp denna funktion och ta emot ett argument i funktionen. Argumentet brukar vi döpa till e och denna gång går det bra att sätta datatyp any :)
i denna funktion skall vi nu göra om texten som vi hämtar från textrutan till ett tal. Detta tal skall sedan lagras i firstNumber
setFirstNumber(parseInt(e.target.value));

12. Kanske behöver göra samma sak för den andra textrutan också.

13. Problemet nu är att vi inte ser någonting i textrutan. Vi behöver sätta varje textrutas värde till dess state-variabel. Detta görs genom html:
<input type='number' onChange={…} value={firstNumber} />

14. Och samma sak för secondNumber.

15. Nu borde du kunna ändra talen och se på skärmen att de ändras. Samtidigt håller dina variables firstNumber och secondNumber reda på vad som står i textrutan så länge du skriver in heltal. Ganska snyggt!

16. Sista delen blir att räkna och skriva ut resultatet. För att kunna använda react på ett bra sätt kommer vi även här att använda oss at state-variabler (egentligen bara en, men

ändå). Skapa upp en till state-variabel som heter result och som har startvärde 0.

17. Skapa en onClick-händelse för knappen och i denna funktion sätter du result-variabeln till firstNumber + secondNumber.

18. Och den sista saken du behöver göra är att presentera result i din html. Do it!