

Reaktiv programmering

Sven A Robbestad
Inmeta Consulting AS

Twitter: @svenardocom

Når noen forsøker å fortelle deg hva
reaktiv programmering er....

Noen klassiske bilder som forsøker å forklare reaktive programmering kommer

1. Introduction
 2. Background
 3. Methodology
 4. Results
 5. Conclusion
 6. References
 7. Appendix
 8. Index
 9. Table of Contents
 10. Summary
 11. Abstract
 12. Keywords
 13. Subject
 14. Author
 15. Date
 16. Page
 17. Chapter
 18. Section
 19. Figure
 20. Table
 21. Equation
 22. Figure
 23. Table
 24. Equation
 25. Figure
 26. Table
 27. Equation
 28. Figure
 29. Table
 30. Equation
 31. Figure
 32. Table
 33. Equation
 34. Figure
 35. Table
 36. Equation
 37. Figure
 38. Table
 39. Equation
 40. Figure
 41. Table
 42. Equation
 43. Figure
 44. Table
 45. Equation
 46. Figure
 47. Table
 48. Equation
 49. Figure
 50. Table
 51. Equation
 52. Figure
 53. Table
 54. Equation
 55. Figure
 56. Table
 57. Equation
 58. Figure
 59. Table
 60. Equation
 61. Figure
 62. Table
 63. Equation
 64. Figure
 65. Table
 66. Equation
 67. Figure
 68. Table
 69. Equation
 70. Figure
 71. Table
 72. Equation
 73. Figure
 74. Table
 75. Equation
 76. Figure
 77. Table
 78. Equation
 79. Figure
 80. Table
 81. Equation
 82. Figure
 83. Table
 84. Equation
 85. Figure
 86. Table
 87. Equation
 88. Figure
 89. Table
 90. Equation
 91. Figure
 92. Table
 93. Equation
 94. Figure
 95. Table
 96. Equation
 97. Figure
 98. Table
 99. Equation
 100. Figure
 101. Table
 102. Equation
 103. Figure
 104. Table
 105. Equation
 106. Figure
 107. Table
 108. Equation
 109. Figure
 110. Table
 111. Equation
 112. Figure
 113. Table
 114. Equation
 115. Figure
 116. Table
 117. Equation
 118. Figure
 119. Table
 120. Equation
 121. Figure
 122. Table
 123. Equation
 124. Figure
 125. Table
 126. Equation
 127. Figure
 128. Table
 129. Equation
 130. Figure
 131. Table
 132. Equation
 133. Figure
 134. Table
 135. Equation
 136. Figure
 137. Table
 138. Equation
 139. Figure
 140. Table
 141. Equation
 142. Figure
 143. Table
 144. Equation
 145. Figure
 146. Table
 147. Equation
 148. Figure
 149. Table
 150. Equation
 151. Figure
 152. Table
 153. Equation
 154. Figure
 155. Table
 156. Equation
 157. Figure
 158. Table
 159. Equation
 160. Figure
 161. Table
 162. Equation
 163. Figure
 164. Table
 165. Equation
 166. Figure
 167. Table
 168. Equation
 169. Figure
 170. Table
 171. Equation
 172. Figure
 173. Table
 174. Equation
 175. Figure
 176. Table
 177. Equation
 178. Figure
 179. Table
 180. Equation
 181. Figure
 182. Table
 183. Equation
 184. Figure
 185. Table
 186. Equation
 187. Figure
 188. Table
 189. Equation
 190. Figure
 191. Table
 192. Equation
 193. Figure
 194. Table
 195. Equation
 196. Figure
 197. Table
 198. Equation
 199. Figure
 200. Table
 201. Equation
 202. Figure
 203. Table
 204. Equation
 205. Figure
 206. Table
 207. Equation
 208. Figure
 209. Table
 210. Equation
 211. Figure
 212. Table
 213. Equation
 214. Figure
 215. Table
 216. Equation
 217. Figure
 218. Table
 219. Equation
 220. Figure
 221. Table
 222. Equation
 223. Figure
 224. Table
 225. Equation
 226. Figure
 227. Table
 228. Equation
 229. Figure
 230. Table
 231. Equation
 232. Figure
 233. Table
 234. Equation
 235. Figure
 236. Table
 237. Equation
 238. Figure
 239. Table
 240. Equation
 241. Figure
 242. Table
 243. Equation
 244. Figure
 245. Table
 246. Equation
 247. Figure
 248. Table
 249. Equation
 250. Figure
 251. Table
 252. Equation
 253. Figure
 254. Table
 255. Equation
 256. Figure
 257. Table
 258. <

I reaktiv programmering er all data en strøm



Joa, alt er en strøm

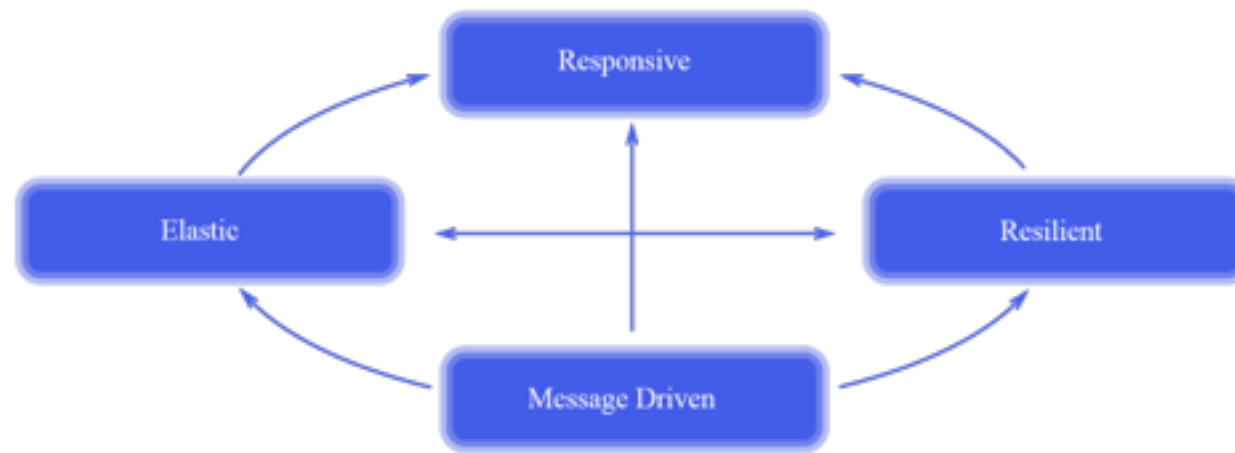
$$x_1(t_{j+1}) = f_1(t_j, x_1, \dots, x_n)$$

\vdots

$$x_n(t_{j+1}) = f_n(t_j, x_1, \dots, x_n)$$

En matematisk tilnærming

Reactive manifesto

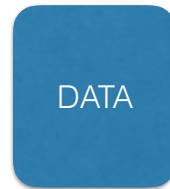


Buzzwords...



Kan ingen bare forklare det på en forståelig måde?

Den enkle sannheten



Observert over tid

I reaktiv programmering observerer du data over tid.

Du observerer et datasett gjennom å abonnere på det.

Når som helst kan datasettet gi fra seg en hendelse. Det kan være et datapunkt, en feilmelding eller en fullført-melding.

Det er alt. I boken fra Gang of four om patterns er reaktiv programmering Observer pattern.



Er ikke dette det samme som hendelsesdrevet
programmering?

Det er mange likhetstrekk. Når du kjenner til prinsippene som driver reaktiv programmering kan du med relativ enkelhet lage et reaktivt funksjons-sett med hendelsesstyrt programmering.



Forskjellen ligger i tankesettet. Med hendelsesdrevet programmering tenker du gjerne at du skal utføre en enkelt handling steg for steg basert på en enkelthendelse. Du tenker imperativt.

Med reaktiv programmering tenker du i stedet på hvordan du skal utføre oppgaven du har fore basert på en innkommende strøm av data. Du bryr deg ikke om hvor disse kommer fra. Du tenker deklarativt.

Reactive rammeverk

RxJS

Bacon.js

Kefir.js

osv.

RXJS finnes også
for Java:

RxJava

Enkel imperativ loop

```
var res;  
for(var i = 0; i<8; i++){  
    if(i % 2 === 0)  
        res = i;  
};  
console.log(`Last even value: ${i}`);  
// -> Last even value: 8
```

Samme kode med reaktiv programmering

```
const range = Rx.Observable.range(1,8)
  .filter((val) => val % 2 === 0)
  .takeLast(1)

range.subscribe(
  (val) => { console.log(`Last even value: ${val}`) }
);
// -> Last even value: 8
```

Forskjellen er her at du frikobler hendelsen med abonnementet på dataene

Eksempel 2

Data fetching med Promises og imperativ loop

```
fetch("http://www.reddit.com/r/puppies.json?limit=5")
.then((response) => response.json())
.then((data) => {
  $('#results')
  .empty()
  var i=0;
  while(i < data.data.children.length) {
    $('#results')
    .append('<div>')
    .append(`${<h3></h3>`.text(data.data.children[i].data.title))
    .append(`${<img>`,{src:data.data.children[i].data.thumbnail}))
    .append('</div>')
    i++;
  }
});
```

Promises

Imperativ kode

Eksempel 2

Data fetching med streams

```
const puppiesStream = Rx.Observable
  .fromPromise(fetch("http://www.reddit.com/r/puppies.json?limit=5"))
  .then((response) => response.json());

puppiesStream.subscribe(
  function(posts) {
    posts.data.children.map((post)=>{
      $('#results')
        .append('<div>')
        .append(`${<h3></h3>`.text(post.data.title))
        .append(`${<img>`,{src:post.data.thumbnail}))
        .append('</div>')
    })
  }
);
```

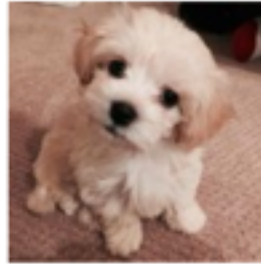
Promise

Reaktiv

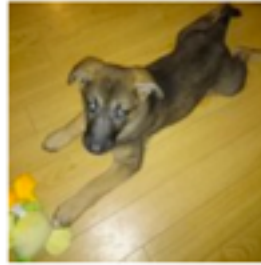
Funksjonell programmering

UI-kode (imperativ)

Qwerty the Maltipoo



My little girl turned eight weeks today!



It's Monday!!



Slutt-resultat

Takk!

Sven A Robbestad
Inmeta Consulting AS

Frontendere ønskes. Ta kontakt med meg om du er
interessert på sverobbe@inmeta.no

Bok om ReactJS og utvikling av ReactJS apps:
<https://www.packtpub.com/web-development/reactjs-blueprints>