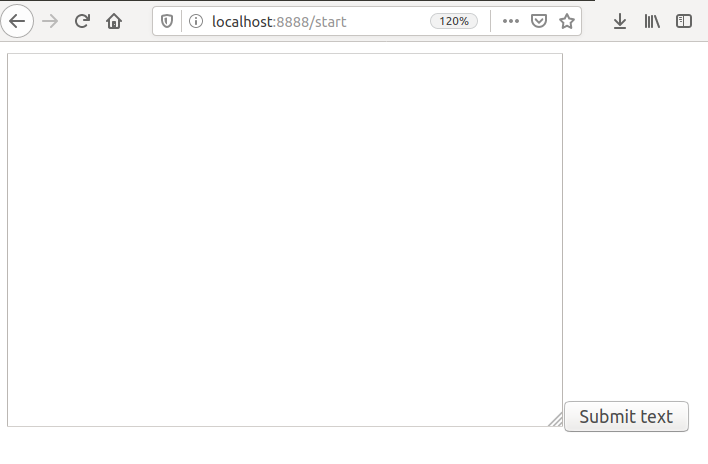
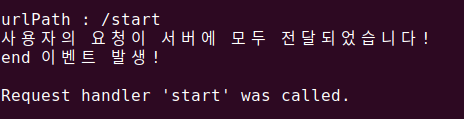
최초에 클라이언트가 ‘/start’ 라는 요청을 하면 start라는 이름의 핸들러 함수가 실행되면서 해당 화면이 나타납니다. 이 때, 요청방식은 당연히 GET방식입니다.

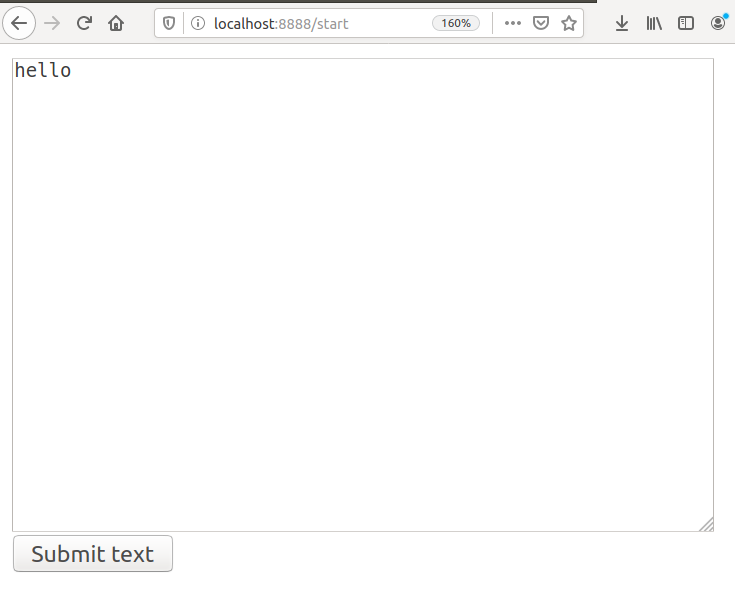


그리고 콘솔창에는 아래와 같은 화면이 출력됩니다.

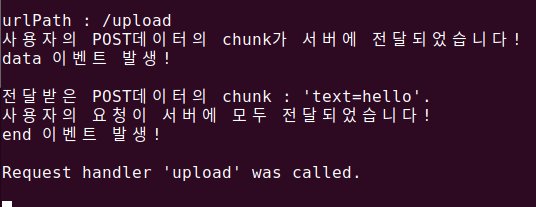
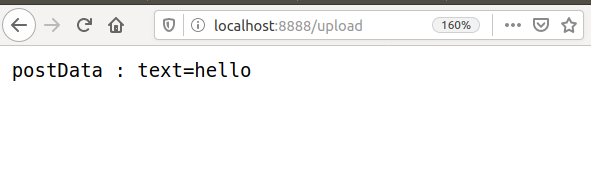


GET방식으로 요청데이터를 전송했기 때문에 별 문제없이 “end”이벤트가 발생하고 start 핸들러 함수를 실행합니다.

이제 클라이언트는 텍스트폼에 어떤 데이터를 입력한 뒤, Submit text버튼을 클릭합니다. “hello”라는 문자열을 포함한 데이터가 POST방식으로 서버에 전달됩니다.



그러면 화면이 바뀌고, 콘솔창에는 아래와 같은 메시지가 출력됩니다.



마지막 사진이 가장 중요합니다. server3.js의 코드와 비교해가면서 확인해보시면 됩니다. 특히 실행 순서를 유심히 보셔야 합니다.

POST방식으로 데이터를 전송했기 때문에 “data” 이벤트가 먼저 여러 번 발생하여(해당 코드에서는 데이터가 크지 않아 한번에 끝났습니다) chunk들을 이어붙인 뒤, 다 붙이면 그제서야 “end” 이벤트가 발생합니다. 그리고 “end” 이벤트에 매핑된 콜백 함수 안에서 route함수를 통해 결과적으로 upload핸들러 함수를 실행합니다.

즉 POST방식의 경우, 여러 번 “data” 이벤트가 발생히여 POST데이터를 온전히 다 받았을 때, “end”이벤트가 발생하여 post데이터를 처리합니다.

아래는 해당 프로그램의 git주소입니다.

<https://github.com/KINGJUSUNG/nodejs/tree/master/ex3>