Node.js

PART II

TIMERS, BUFFER, STREAMS, TCP

so far we learnt to …

- **use nodejs to run js on desktop**

- create our own modules
- about asynchronous callbacks
- emit events and handle events

Goals for this session

1. show Buffer class
2. show how to create a simple server
3. show how to create a webserver
4. show how to create websockets

Buffer

- Javascript useful for handling strings.
- How about binary data (such as images etc).
- Buffer class provided for easy handling of binary data (such as images etc)

Encodings

To store a unicode character (example: ö)

- needs less than a byte of base64

  (ö cannot be represented in base64)
- needs 1 byte of ascii (only a subset of unicode)

  (ö cannot be represented in base64)
- needs between 1 and 4 bytes of UTF-8 encoding

  (ö is represented as c3 b6 in UTF-8)
- needs between 1 and 2 UTF-16 encoding

  (ö is represented as c3b6 in UTF-16)

base64

This is the base64 mapping table.

0 A	17 R	34 i	51 z
1 B	18 S	35 j	52 0
2 C	19 T	36 k	53 1
3 D	20 U	37 l	54 2
4 E	21 V	38 m	55 3
5 F	22 W	39 n	56 4
6 G	23 X	40 o	57 5
7 H	24 Y	41 p	58 6
8 I	25 Z	42 q	59 7
9 J	26 a	43 r	60 8
10 K	27 b	44 s	61 9
11 L	28 c	45 t	62 +
12 M	29 d	46 u	63 /
13 N	30 e	47 v	
14 O	31 f	48 w	
15 P	32 g	49 x	
16 Q	33 h	50 y	

UTF-8

U+005A	Z	5a	LATIN CAPITAL LETTER Z
U+005B	[	5b	LEFT SQUARE BRACKET
U+005C	\	5c	REVERSE SOLIDUS
U+005D	]	5d	RIGHT SQUARE BRACKET
U+005E	^	5e	CIRCUMFLEX ACCENT
U+005F	_	5f	LOW LINE
U+0060	`	60	GRAVE ACCENT

U+00A0		c2 a0	NO-BREAK SPACE
U+00A1	¡	c2 a1	INVERTED EXCLAMATION MARK
U+00A2	¢	c2 a2	CENT SIGN
U+00A3	£	c2 a3	POUND SIGN
U+00A4	¤	c2 a4	CURRENCY SIGN
U+00A5	¥	c2 a5	YEN SIGN
U+00A6	¦	c2 a6	BROKEN BAR
U+00A7	§	c2 a7	SECTION SIGN

Example

```
var buf = new Buffer("Hello World", "ascii"); // ascii is 8 bit
//var buf = new Buffer("Hello World");

var buf = new Buffer("Kölniअे"); // utf-8 is a multi-byte encoding
console.log(buf);
//<Buffer 4b |c3 b6| 6c 6e 69 |e0 a4 84|>

buf = new Buffer("8b76fde713ce",'base64');
// base64 is 6-bit encoding


console.log(buf);
//<Buffer f1 be fa 7d d7 bb d7 77 1e>
```

Servers

Simple servers

```
require('net')

createServer()
listen(port#)
'error'
'connection'
'data'
'close'
```

HTTP servers

```
require('http')

createServer()
listen(port#)
'request'
    req.on 'data'
```