ByteString/Data Type (#6361)

- In Ruby 1.8, Strings were used to store textual data and arbitrary binary data. This was possible because bytes were bytes...
- In Ruby 1.9, Strings always have encoding data, and bytes are codepoints, but people still try to use Strings to store arbitrary binary data.
- This causes a problem:

```
> a = "test".force_encoding('BINARY')
> b = "\xFF".force_encoding('BINARY')
> a << "t\xC3\xA9st"
"testtést"
> b << "t\xC3\xA9st"
Encoding::CompatibilityError: incompatible character encodings:
ASCII-8BIT and UTF-8
```

=> The only way to reliably use Strings to store binary data is to always `#force_encoding(‘BINARY’)` on every string. This is annoying and inconvenient.

Please consider a ByteString/Data type:

- Array’s store objects, and bytes are not objects
- Strings always have encoding, but bytes don’t always have encoding (i.e. bytes read from a socket)
- A Data type can store arbitrary collections of bytes

Other benefits:

- `String#getbyte`/`String#setbyte` become `Data#[]`/`Data#[]=`
- Easy to apply byte-masks:

```
a = “\x33\x44\x55”.to_data; b = “\x00\x00\xFF”.to_data; a | b #=> “\x00\x00\x55”
```