Scala.js

Scalable, maintainable web apps

Paul Draper

- Senior software engineer at Lucid Software
- Full-stack developer
- Salt Lake City, UT

Scalability

What is “scalabilty”?

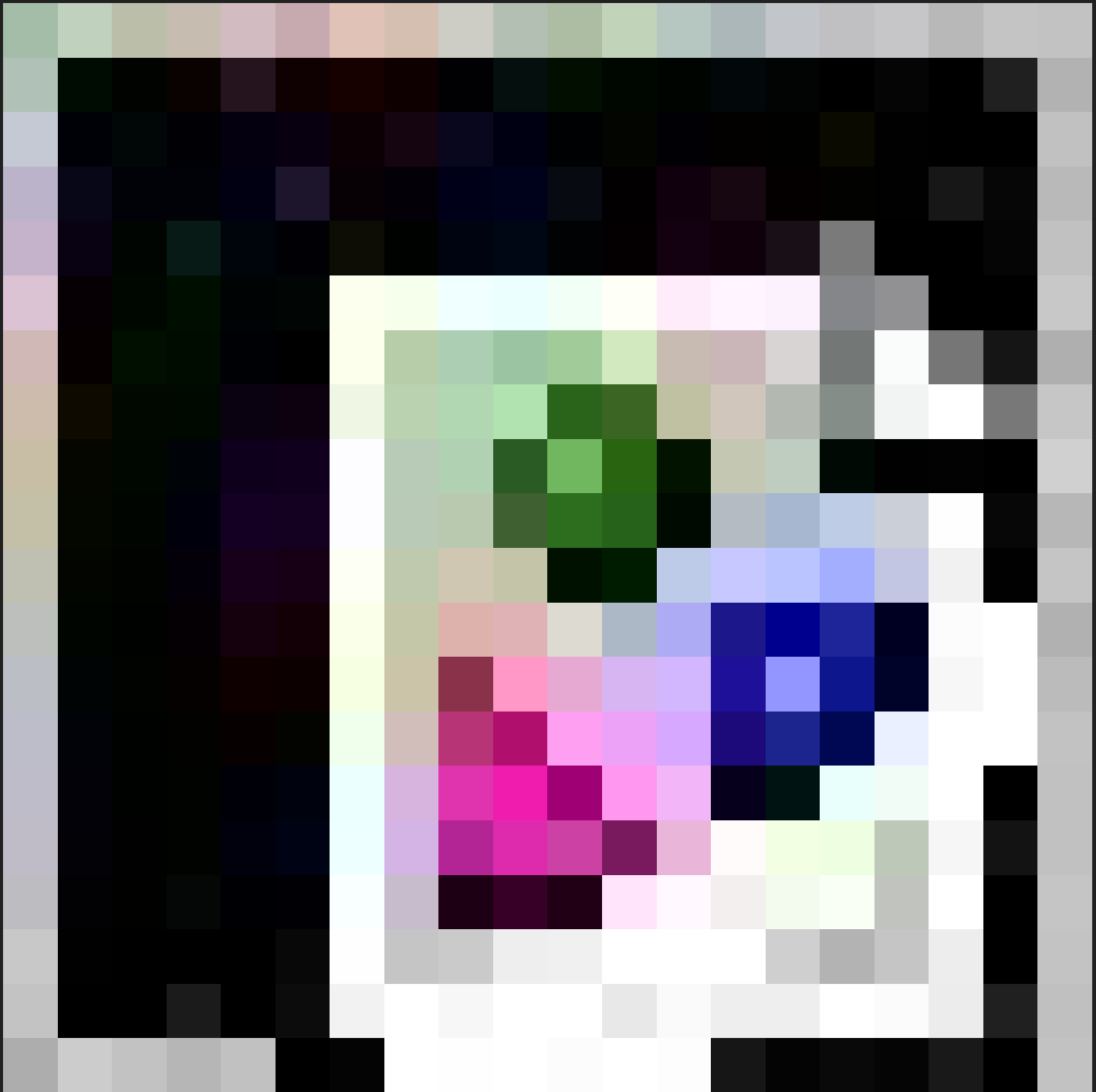
The ability to change size or capacity without compromising efficiency or functionality

Scaling what?

- Users
- Persistent data
- Number of tasks
- Size of tasks
- Revenue

Scaling what?

Technical complexity



Lucidchart

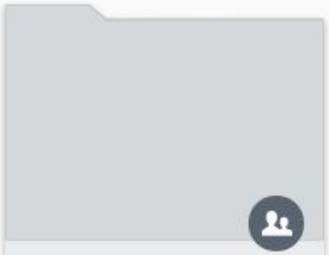
- My Documents
  - <>
  - A folder
- New Folder

- Recent Documents
- Shared with Me (26)
- Trash

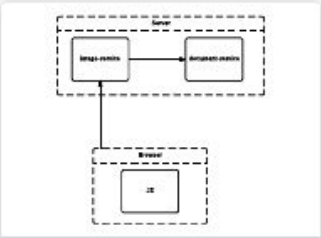
Need a layout & design tool?
Try Lucidpress

How are we doing?

+ Document | + Folder | Import

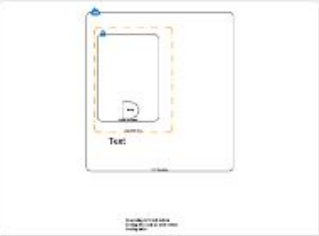

<>

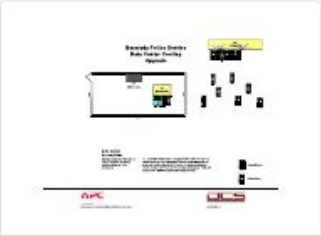
A folder


Blank Flowchart

Blank Flowchart


AWS Import - Lucidchart


Blank Flowchart


Bermuda Police Service

Blank Flowchart

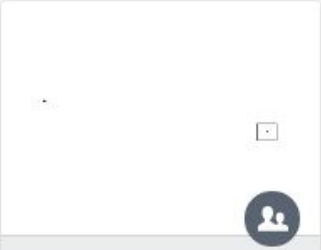
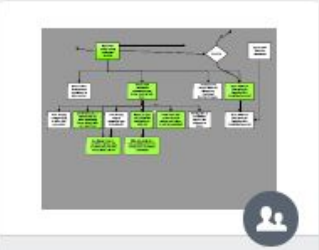
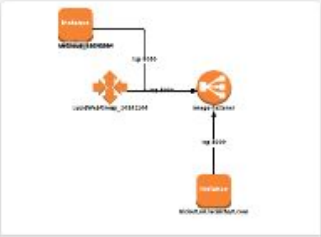


CHART-8747 Bullets


Google Drive integration


Blank Flowchart


Blank Flowchart

Download

FORMAT

- PDF (Recommended for printing)
- PNG (Recommended image format)
- PNG with transparent background
- JPEG
- Visio (VDX)
- SVG
- SVG with transparent background

CONTENT

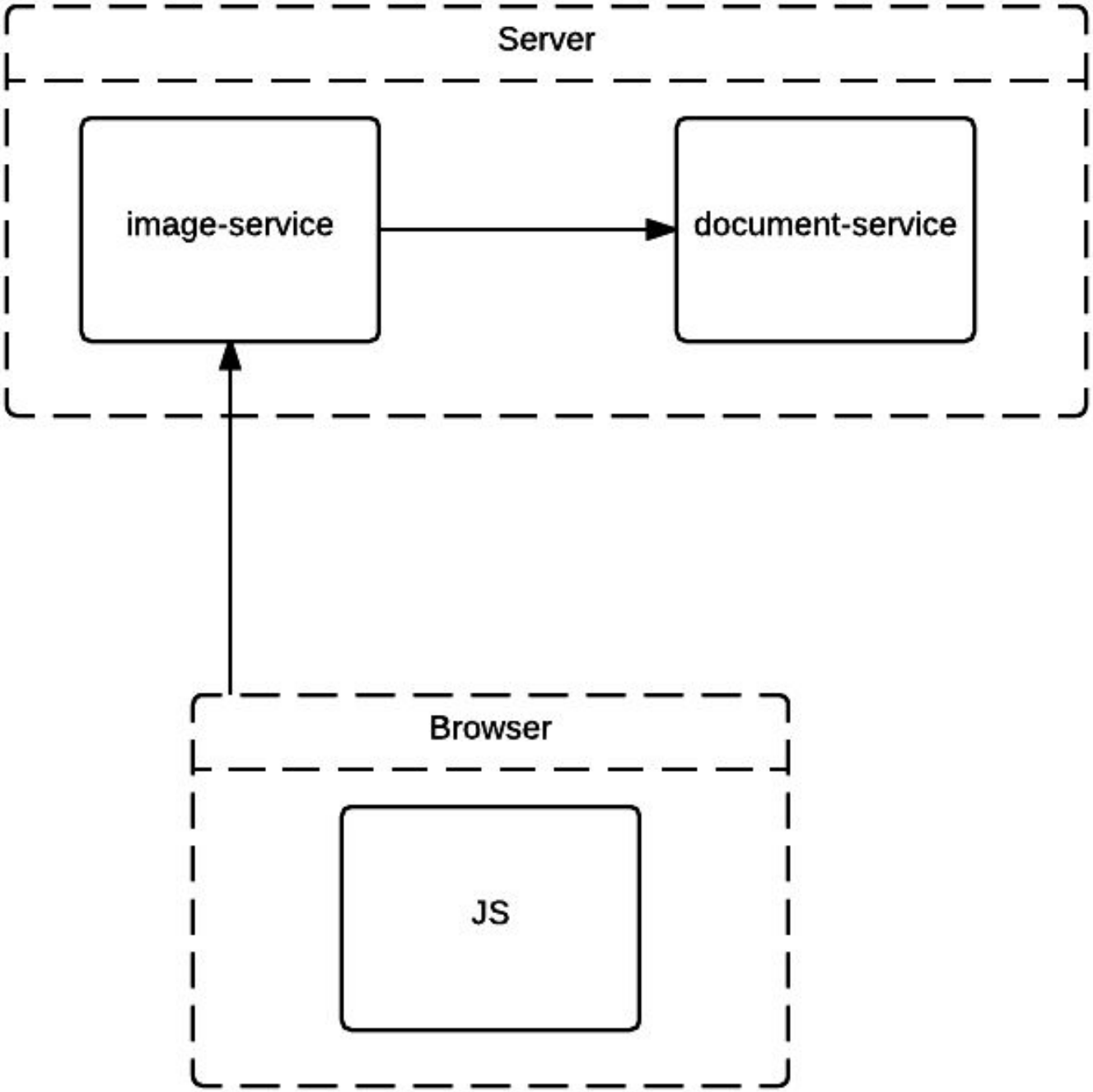
- Document
- Document with Layers
- Page Range e.g. 1-5,8,11-13
- Current Page
- Crop to Content
- Part of Current Page

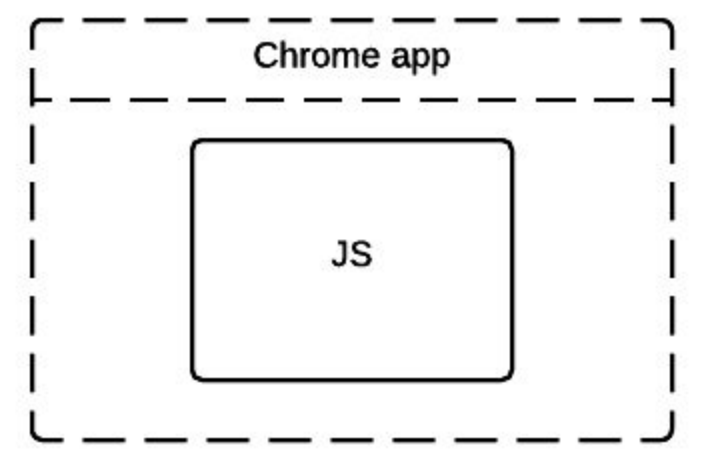
QUALITY

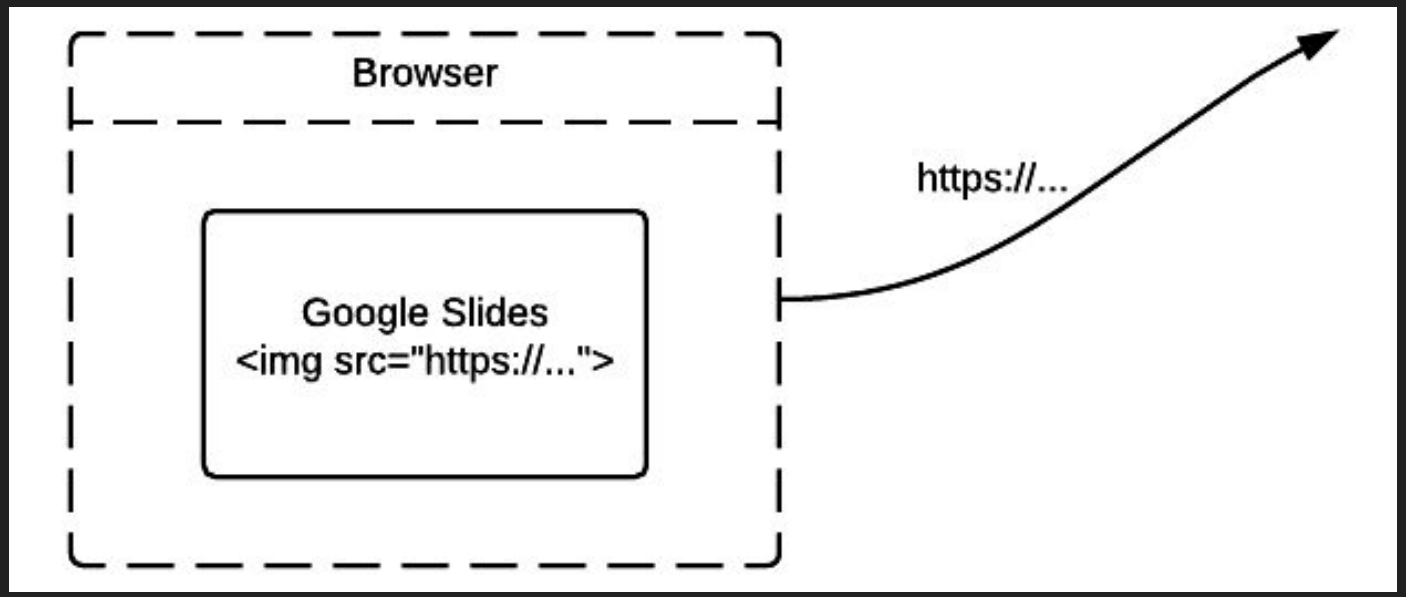
- Current Zoom Level
- Screen Quality (160 DPI)
- Print-quality (300 DPI)

Download

Where do we put the code?

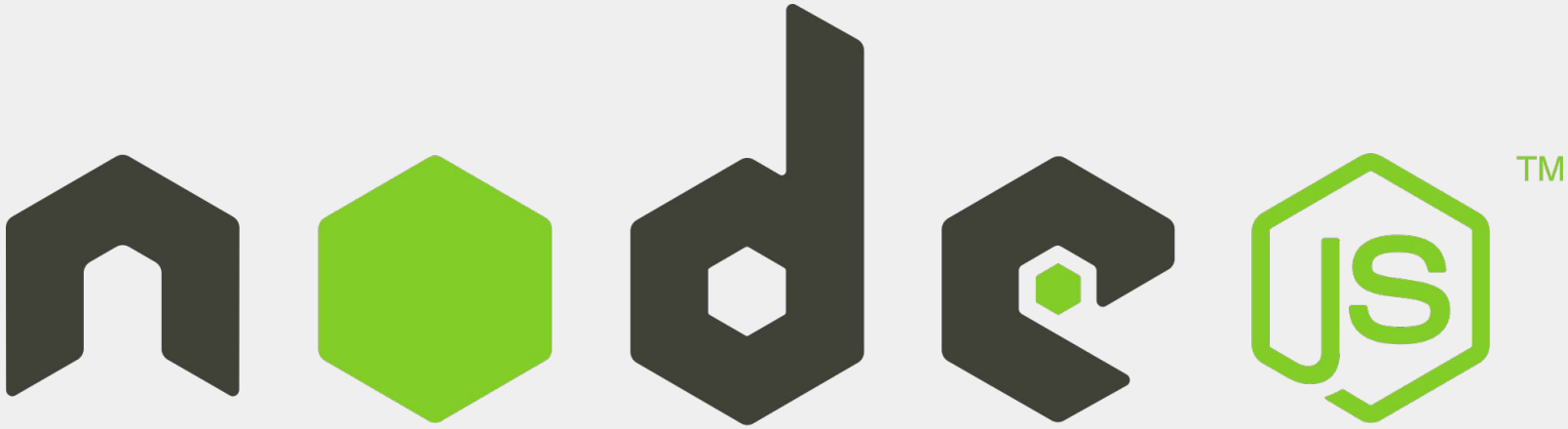

Server
image-service
document-service
Browser
JS


Chrome app
JS




```
<img
    src="https://documents.lucidchart.com/documents/1d170967-4768-4738-bbeb-
    9d3bcba7309e/pages/at01Y-4qI6-e?
    a=393&x=492&y=128&w=627&h=264&store=1&accept=image%2F*&auth=LCA%
    2030c40c3d3be121c4fd52ec7fb71fe0532afc1147-ts%3D1457544344"
>
```

HTML
CSS
JS


node
JS
TM

JavaScript is built on some very good ideas and a few very bad ones.

– Douglas Crockford, *JavaScript: The Good Parts*

JavaScript

– Two kinds of null

```
undefined   null
```

– Three kinds of equals

```
==   ===   Object.is
```

– Three ways for functions (and two ways for `this`)

```
var f = function(){};   function f(){}   var f = ()=>undefined
```

– Weak typing

```
'5' + '5' - '5'
```

– Non-transitive comparisons

```
'2' < 5    5 < '10'    '10' < '2'
```

– Confusing types

```
1 vs. new Number(1)
```

JavaScript

Many issues can be worked around with linters and collective experience.

Other issues will likely never be solved, e.g. shared-memory CPU parallelism.

How often do you need to use more processors? (I.e. scale)

Compile-to-JS

https://github.com/jashkenas/coffeescript/wiki/list-of-languages-that-compile-to-js

Scalable + Language

History of Scala

- 2001 - initial Scala design by Martin Odersky at EPFL (Swiss)
- 2003 - internal release
- 2004 - public release, first JVM and then .NET
- 2006 - Scala v2
- 2011 - Typesafe Inc.
- 2013 - Scala.js prototype, followed by public version
- 2014 - Scala.js passes Scala test suite, gets incremental building

Why Scala?

Scalability

Why Scala?

- High-level – developer efficiency
- Functional – easy to reason through
- Static types – catch errors earlier
- Type inference – easier to read and write
- Concurrency – leverage hardware abilities
- Macros – replace need for runtime dynamism and reflection
- Performance – JIT
- Runtime ubiquity – JVM and JavaScript

Type safety is the extent to which a programming language discourages or prevents type errors.

https://wikipedia.org/wiki/Type_safety

Example class

```
case class Person(name: String, birthdate: Date)


class Person(name, birthdate) { // ES6
  constructor(name, birthdate) {
    this.name = name;
    this.birthdate = birthdate;
  }
}
```

Collections

```
Seq(1, 2, 3)

1 :: 2 :: 3 :: Nil

Map('a -> 1, 'b -> 2, 'c -> 3)

ListMap(1 -> "b", 2 -> "a")

Set(new Object, new Object)

TreeSet("a", "b")

BitSet(1, 2, 3)
```

Collections

```
val strings = Seq("ed", "vim", "emacs")
val (short, long) = strings.partition(_.size < 3)
> short: Seq("ed")
> long : Seq("vim", "emacs")
```

Monads

```
val present: Option[String] = Some("Lucid Software")

present.map(_.toLowerCase)

> Some("lucid software")

val absent: Option[String] = None

absent.map(_.toLowerCase)

> None

present flatMap {
  case "" => None
  case x => x + "!"
}

> Some("Lucid Software")
```

Monads

```
val success: Either[String, Seq[Int]] = Right(Seq(1, 2))

success.right.map(_.reverse)

> Right(Seq(2, 1))

val failure: Either[String, Seq[Int]] = Left("Invalid input")

failure.right.map(_.reverse)

> Left("Invalid input")
```

Pattern matching

```
val list = List(List(1, 2), List(3))


list match {
  case List(a, 2) +: rest => a
  case first +: rest if first.nonEmpty => rest
  case _ => Nil
}
```

String interpolation

```
val username = "paul"
s"$username@lucidchart.com"

val name = "Robert'); DROP TABLE Students; --"
sql"SELECT * FROM Students WHERE name = $name"

val data = "<script>alert()</script>"
html"<div>$data</div>"
```

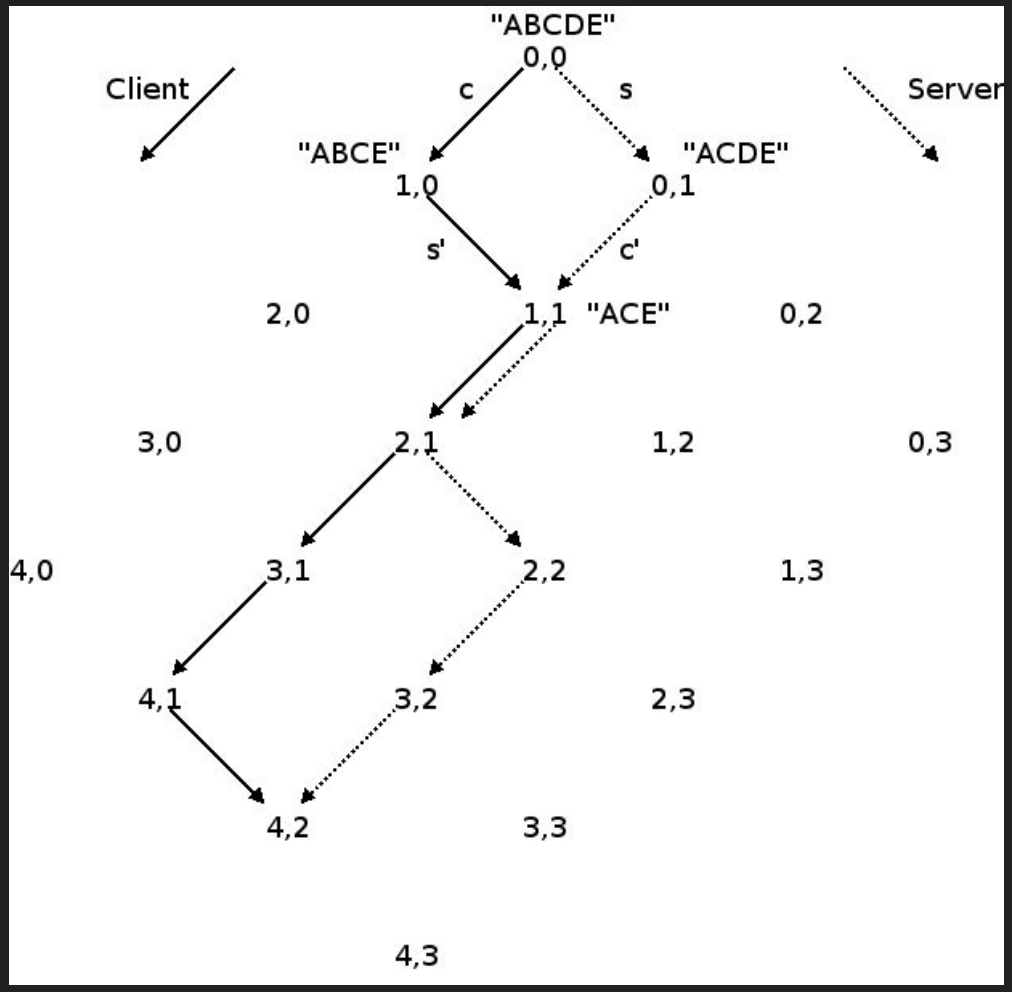
DSLs

```
object Lunar {
  def main(args: Array[String): Unit = {
    10 PRINT "Welcome to Baysick Lunar Lander v0.9"
    20 LET ('dist := 100)
    30 LET ('v := 1)
    40 LET ('fuel := 1000)
    50 LET ('mass := 1000)
    60 PRINT "You are drifting towards the moon."
  }
}
```

Macros

```
case class User(name: String, hobbies: Seq[Hobby])

case class Hobby(name: String, priority: Int)


val user = User("john", Seq(Hobby("skiing", 1)))

val json = Pickle.intoString(user)

> {"name": "John", "hobbies": [{"name": "skiing", priority: 1}]}

Unpickle.fromString(json)

> Success(User("john", Seq(Hobby("skiing", 1)))
```

Example: operational transforms




I CAN HAZ
LIBRARIES?
imgflip.com

Dynamic calls

```
import js.Dynamic._


val elements = global.$(":hidden")

elements.show()

elements.initTerribleCarousel();


newInstance(global.MyFunction); // new MyFunction
```

Facade

```
@js.native
trait Window extends js.Object {
  val document: HTMLDocument = js.native
  var location: String = js.native

  def innerWidth: Int = js.native
  def innerHeight: Int = js.native

  def alert(message: String): Unit = js.native

  def open(url: String, target: String,
      features: String = ""): Window = js.native
  def close(): Unit = js.native
}
```

Libraries

https://github.com/scala-js/scala-js-dom

https://github.com/scala-js/scala-js-jquery

https://github.com/greencatsoft/scalajs-angular

https://github.com/japgolly/scalajs-react

Libraries

https://github.com/sjrd/scala-js-actors

https://github.com/milessabin/shapeless

https://github.com/benhutchison/prickle

Questions?

Paul Draper
paul@lucidchart.com