accenturetechnology

PROFESSIONAL EDUCATION

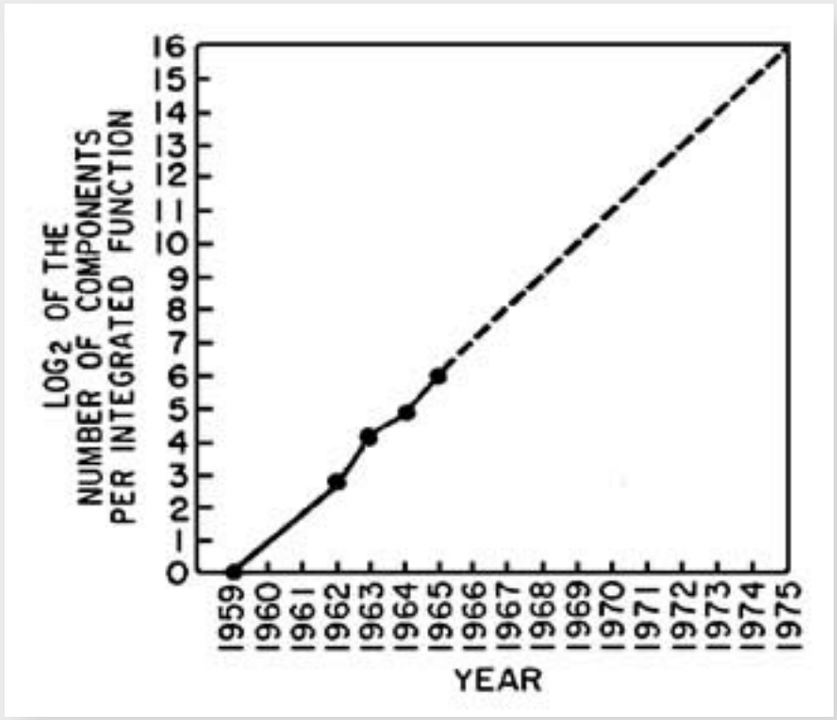
ACCENTURE TECHNOLOGY ACADEMY 2.0

LOW CODE, CITIZEN DEVELOPMENT & NEW PROGRAMMING PARADIGMS

DANIEL JACKSON

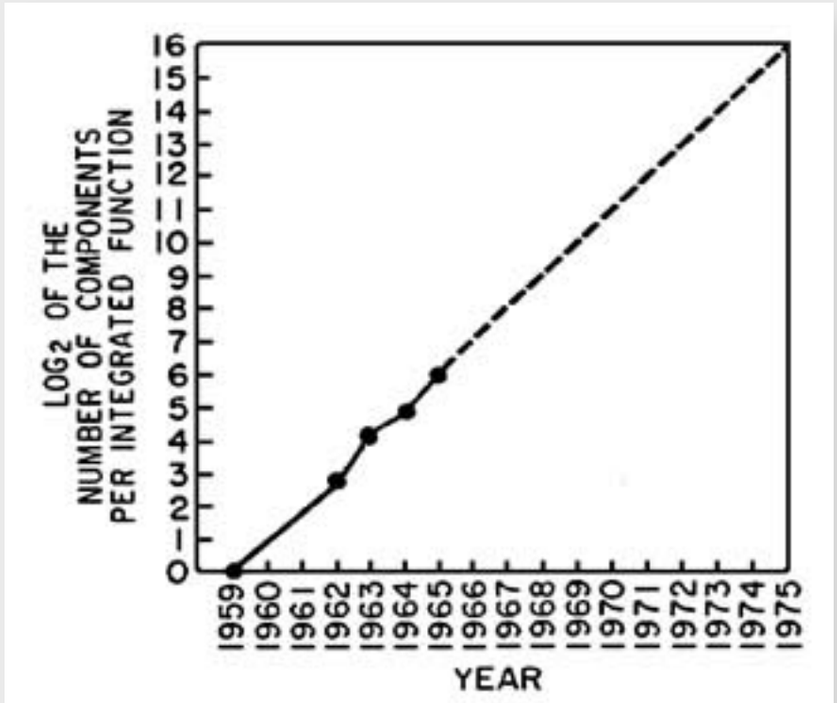
the growth of computational power

more transistors

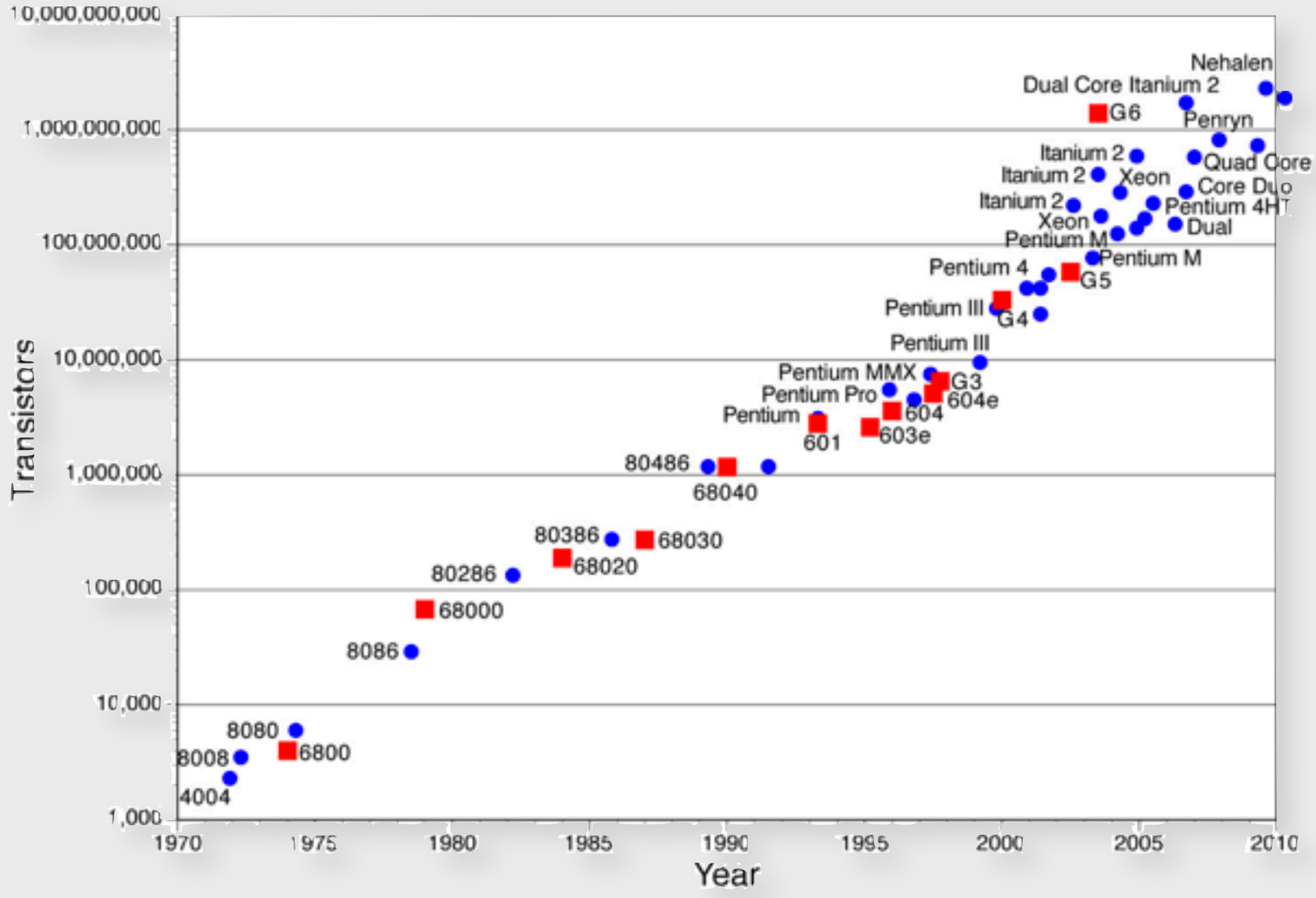


predicted (1965)

more transistors

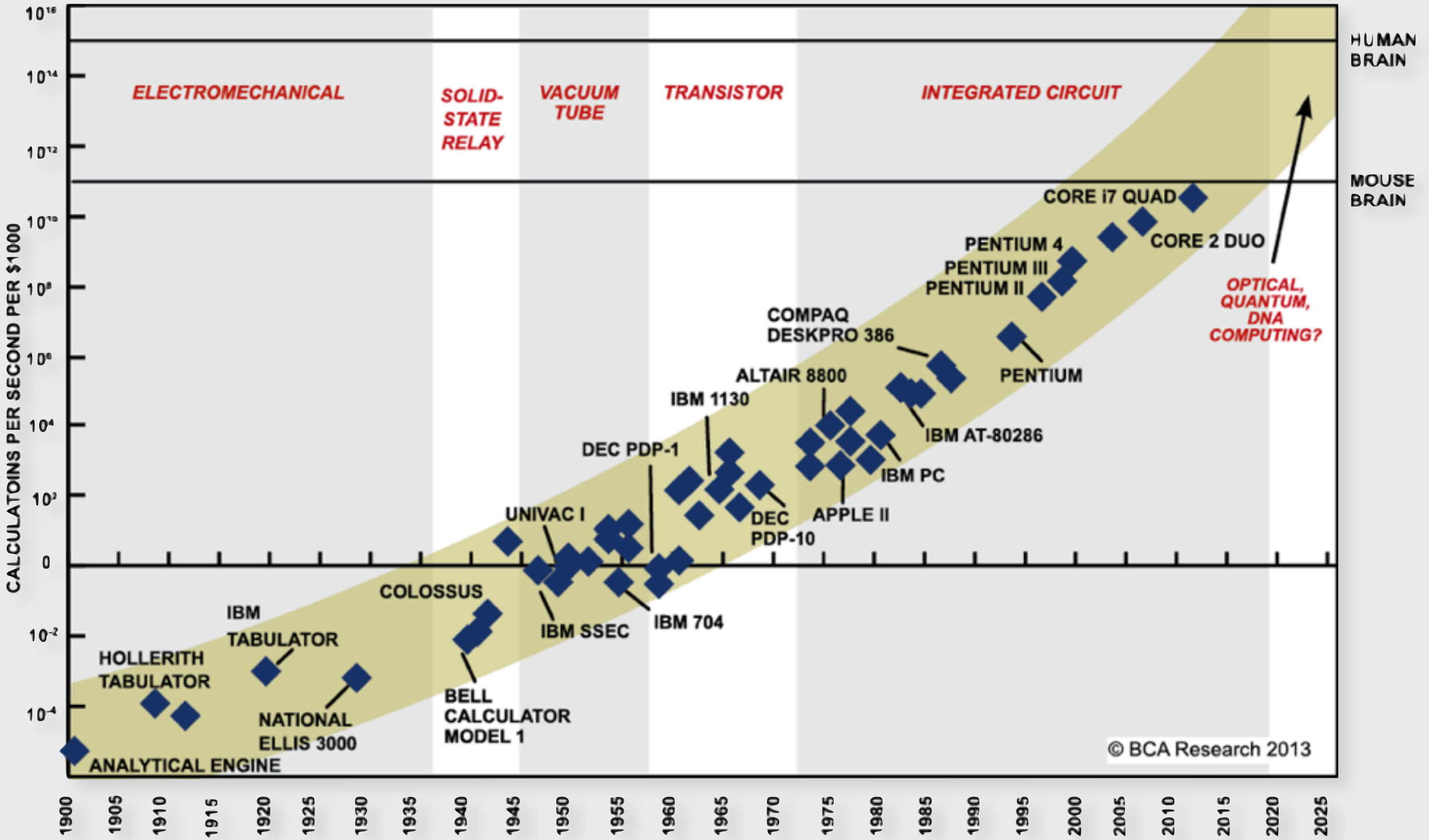


predicted (1965)



actual (2010)

more calculations



SOURCE: RAY KURZWEIL, "THE SINGULARITY IS NEAR: WHEN HUMANS TRANSCEND BIOLOGY", P.67, *THE VIKING PRESS*, 2006. DATAPOINTS BETWEEN 2000 AND 2012 REPRESENT BCA ESTIMATES.

reversing a string, 1960-1990

```
REVERSE  CSECT
         USING  REVERSE,R13
         B      72(R15)
         DC     17F'0'
         STM    R14,R12,12(R13)
         ST     R13,4(R15)
         ST     R15,8(R13)
         LR     R13,R15
         MVC    TMP(L'C),C
         LA     R8,C
         LA     R9,TMP+L'C-1
         LA     R6,1
         LA     R7,L'C
LOOPI    CR     R6,R7
         BH     ELOOPI
         MVC    0(1,R8),0(R9)
         LA     R8,1(R8)
         BCTR   R9,0
         LA     R6,1(R6)
         B      LOOPI
ELOOPI   XPRNT  C,L'C
         L      R13,4(0,R13)
         LM     R14,R12,12(R13)
         XR     R15,R15
         BR     R14
C        DC     CL12'edoC attesoR'
TMP      DS     CL12
         YREGS
         END    REVERSE
```

360 Assembly

reversing a string, 1960-1990

```
REVERSE  CSECT
         USING  REVERSE,R13
         B      72(R15)
         DC     17F'0'
         STM    R14,R12,12(R13)
         ST     R13,4(R15)
         ST     R15,8(R13)
         LR     R13,R15
         MVC    TMP(L'C),C
         LA     R8,C
         LA     R9,TMP+L'C-1
         LA     R6,1
         LA     R7,L'C
LOOPI    CR     R6,R7
         BH     ELOOPI
         MVC    0(1,R8),0(R9)
         LA     R8,1(R8)
         BCTR   R9,0
         LA     R6,1(R6)
         B      LOOPI
ELOOPI   XPRNT  C,L'C
         L      R13,4(0,R13)
         LM     R14,R12,12(R13)
         XR     R15,R15
         BR     R14
C        DC     CL12'edoC attesoR'
TMP      DS     CL12
         YREGS
         END    REVERSE
```

360 Assembly

```
PROGRAM Example

  CHARACTER(80) :: str = "This is a string"
  CHARACTER :: temp
  INTEGER :: i, length

  WRITE (*,*) str
  length = LEN_TRIM(str)
  DO i = 1, length/2
     temp = str(i:i)
     str(i:i) = str(length+1-i:length+1-i)
     str(length+1-i:length+1-i) = temp
  END DO
  WRITE(*,*) str

END PROGRAM Example
```

Fortran

reversing a string, 1960-1990

```
REVERSE  CSECT
         USING  REVERSE,R13
         B      72(R15)
         DC     17F'0'
         STM    R14,R12,12(R13)
         ST     R13,4(R15)
         ST     R15,8(R13)
         LR     R13,R15
         MVC    TMP(L'C),C
         LA     R8,C
         LA     R9,TMP+L'C-1
         LA     R6,1
         LA     R7,L'C
LOOPI    CR     R6,R7
         BH     ELOOPI
         MVC    0(1,R8),0(R9)
         LA     R8,1(R8)
         BCTR   R9,0
         LA     R6,1(R6)
         B      LOOPI
ELOOPI   XPRNT  C,L'C
         L      R13,4(0,R13)
         LM     R14,R12,12(R13)
         XR     R15,R15
         BR     R14
C        DC     CL12'edoC attesoR'
TMP      DS     CL12
         YREGS
         END    REVERSE
```

360 Assembly

```
PROGRAM Example

  CHARACTER(80) :: str = "This is a string"
  CHARACTER :: temp
  INTEGER :: i, length

  WRITE (*,*) str
  length = LEN_TRIM(str)
  DO i = 1, length/2
     temp = str(i:i)
     str(i:i) = str(length+1-i:length+1-i)
     str(length+1-i:length+1-i) = temp
  END DO
  WRITE(*,*) str

END PROGRAM Example
```

Fortran

```
PROC reverse = (REF STRING s)VOID:
  FOR i TO UPB s OVER 2 DO
    CHAR c = s[i];
    s[i] := s[UPB s - i + 1];
    s[UPB s - i + 1] := c
  OD;
```

Algol 68

reversing a string, 1960-1990

```
REVERSE  CSECT
         USING  REVERSE,R13
         B      72(R15)
         DC     17F'0'
         STM    R14,R12,12(R13)
         ST     R13,4(R15)
         ST     R15,8(R13)
         LR     R13,R15
         MVC    TMP(L'C),C
         LA     R8,C
         LA     R9,TMP+L'C-1
         LA     R6,1
         LA     R7,L'C
LOOPI    CR     R6,R7
         BH     ELOOPI
         MVC    0(1,R8),0(R9)
         LA     R8,1(R8)
         BCTR   R9,0
         LA     R6,1(R6)
         B      LOOPI
ELOOPI   XPRNT  C,L'C
         L      R13,4(0,R13)
         LM     R14,R12,12(R13)
         XR     R15,R15
         BR     R14
C        DC     CL12'edoC attesoR'
TMP      DS     CL12
         YREGS
         END    REVERSE
```

360 Assembly

```
PROGRAM Example

  CHARACTER(80) :: str = "This is a string"
  CHARACTER :: temp
  INTEGER :: i, length

  WRITE (*,*) str
  length = LEN_TRIM(str)
  DO i = 1, length/2
     temp = str(i:i)
     str(i:i) = str(length+1-i:length+1-i)
     str(length+1-i:length+1-i) = temp
  END DO
  WRITE(*,*) str

END PROGRAM Example
```

Fortran

```
PROC reverse = (REF STRING s)VOID:
  FOR i TO UPB s OVER 2 DO
    CHAR c = s[i];
    s[i] := s[UPB s - i + 1];
    s[UPB s - i + 1] := c
  OD;
```

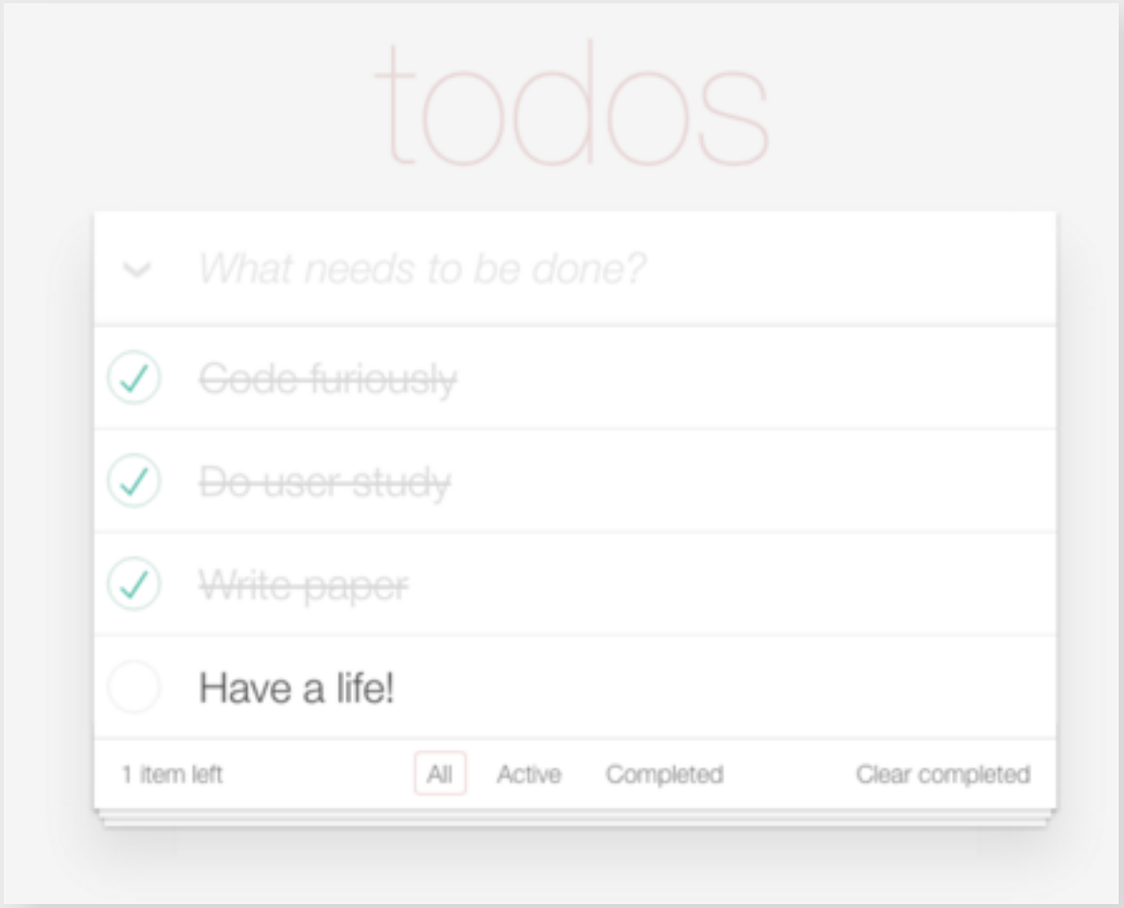
Algol 68

```
reverse = foldl (flip (:)) []
```

Haskell

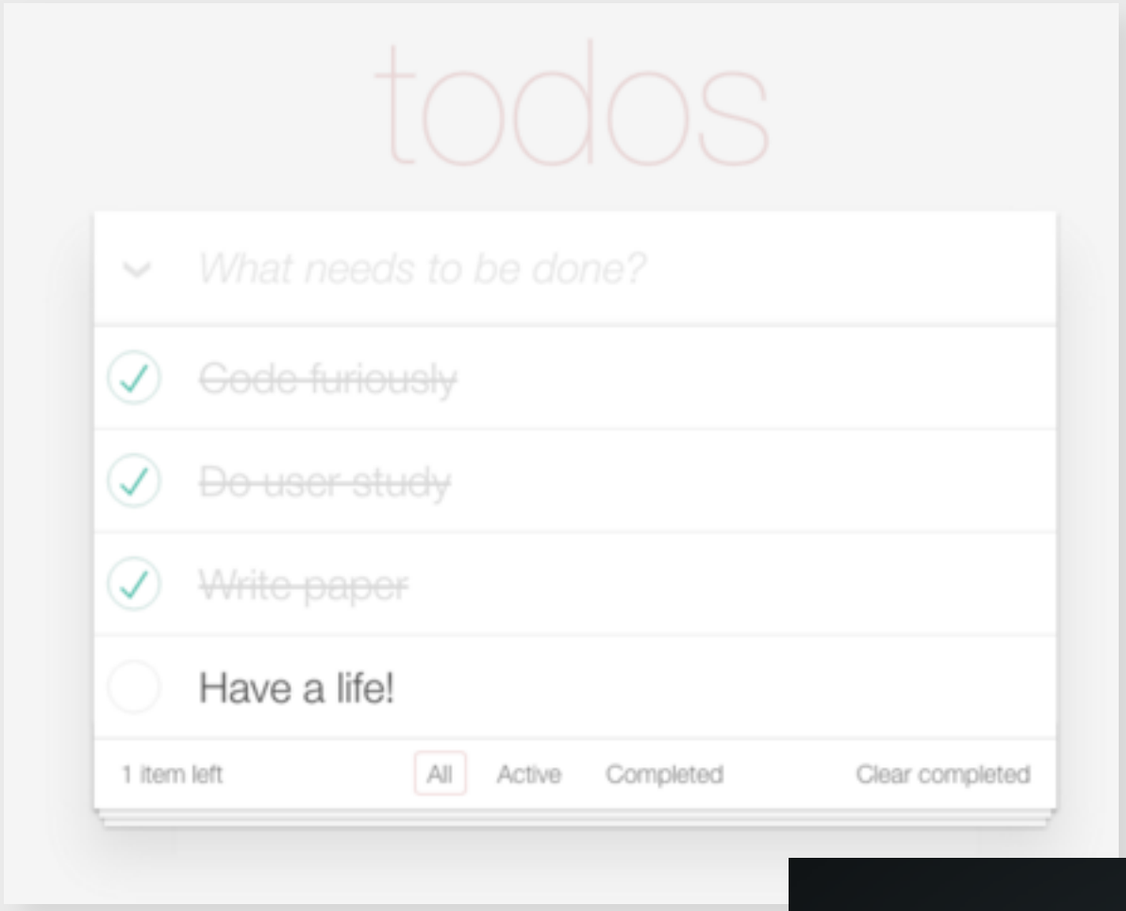
a benchmark example

todomvc.com
showcase of MVC
frameworks



a benchmark example

todomvc.com
showcase of MVC frameworks

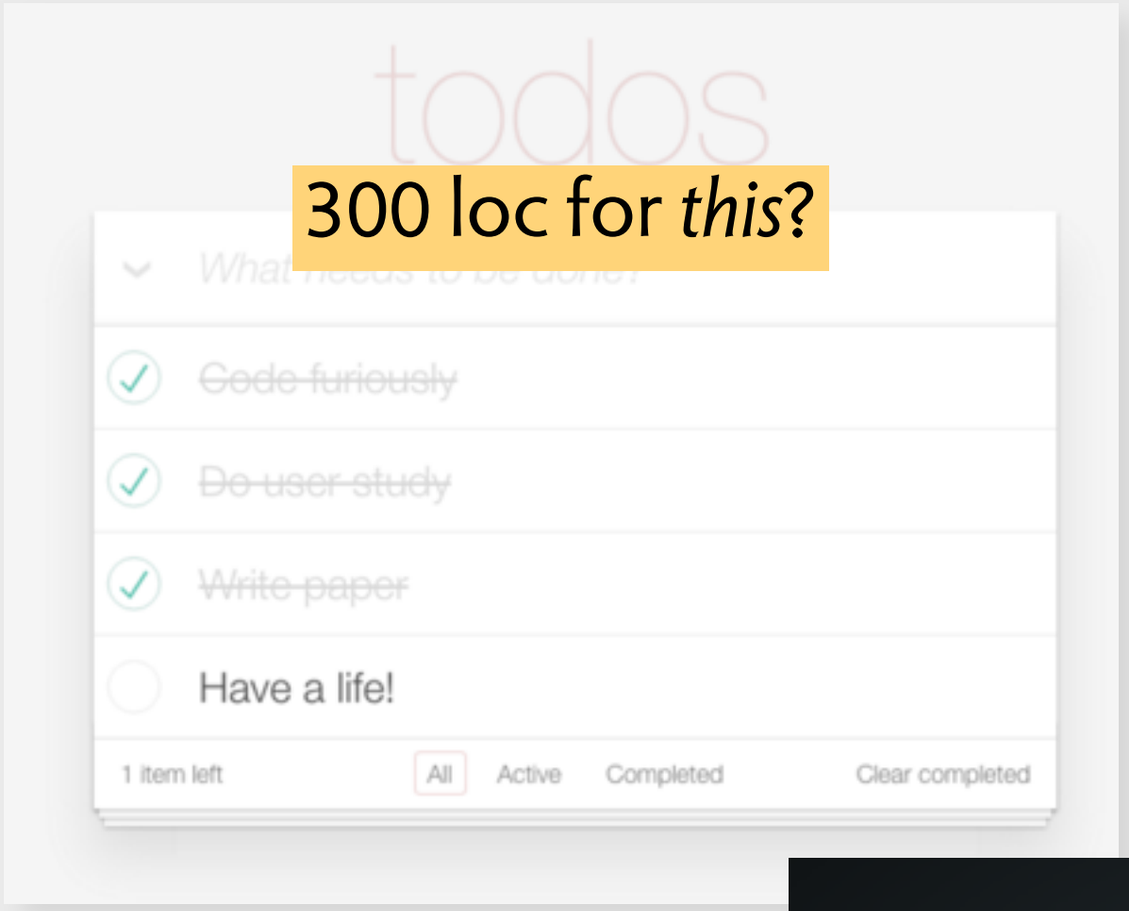


todomvc.com

Framework	SLOC
Angular	294
Polymer	246
React	421
Backbone	297

a benchmark example

todomvc.com
showcase of MVC frameworks

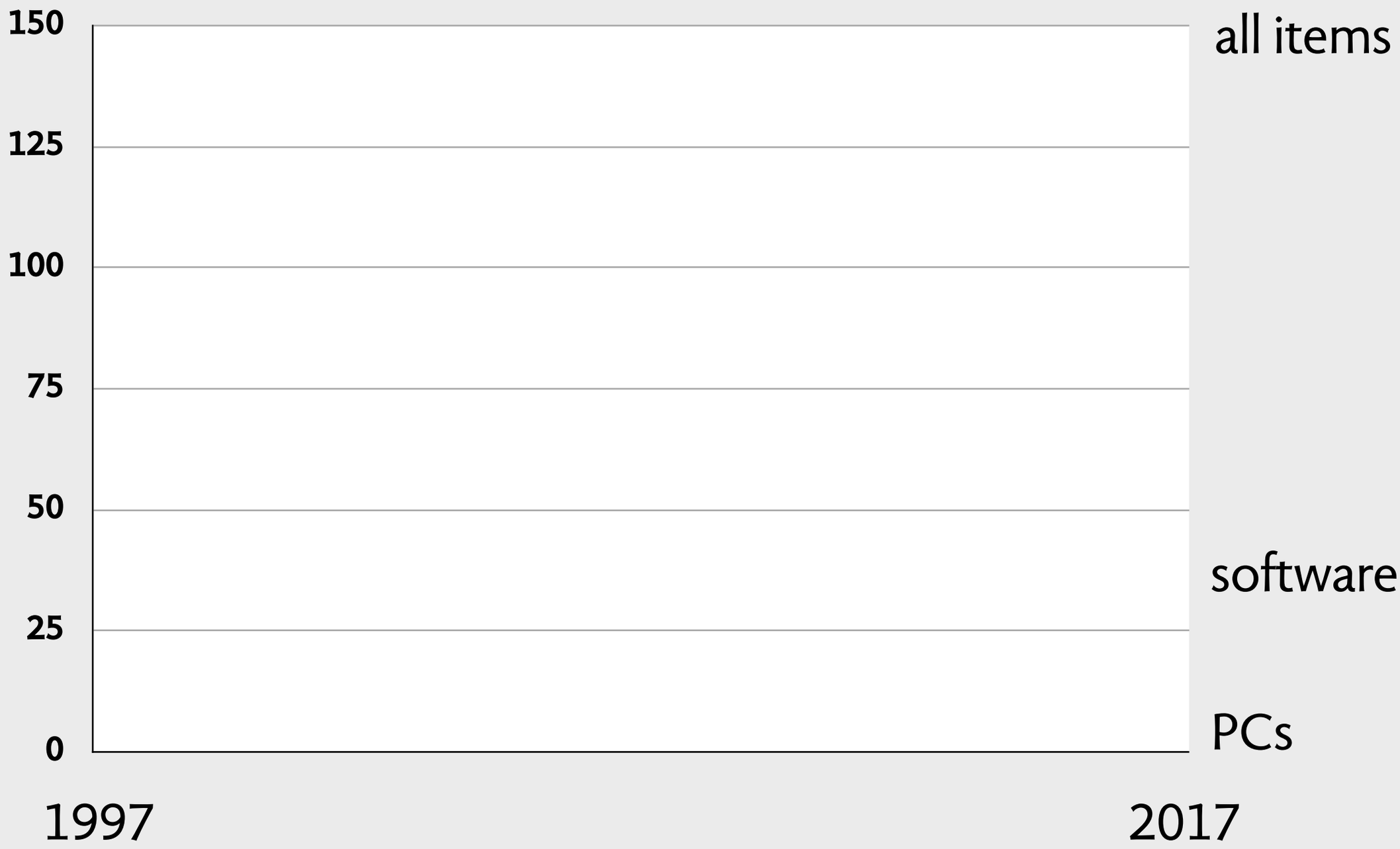


todomvc.com

Framework	SLOC
Angular	294
Polymer	246
React	421
Backbone	297

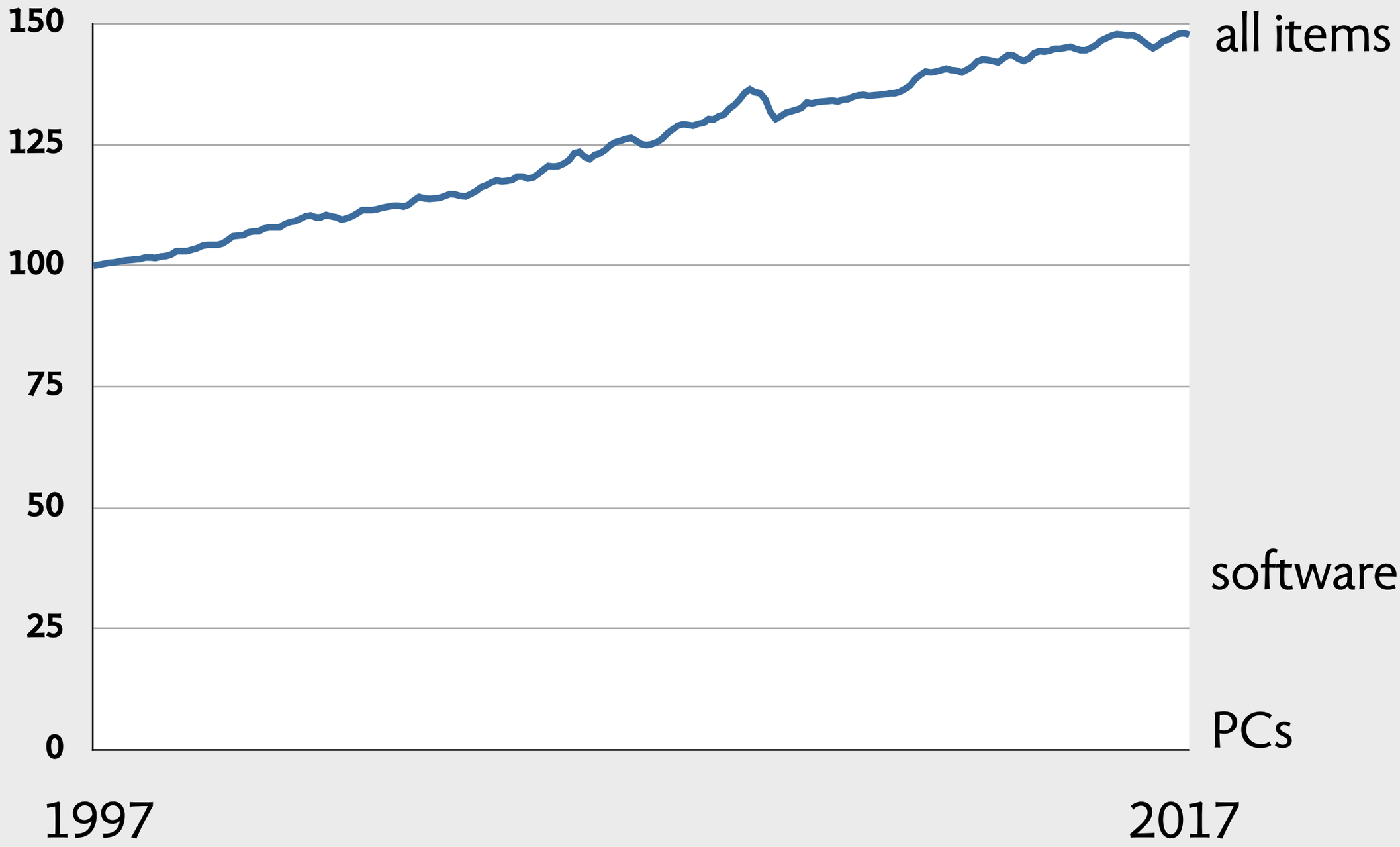
pain points

consumer price index



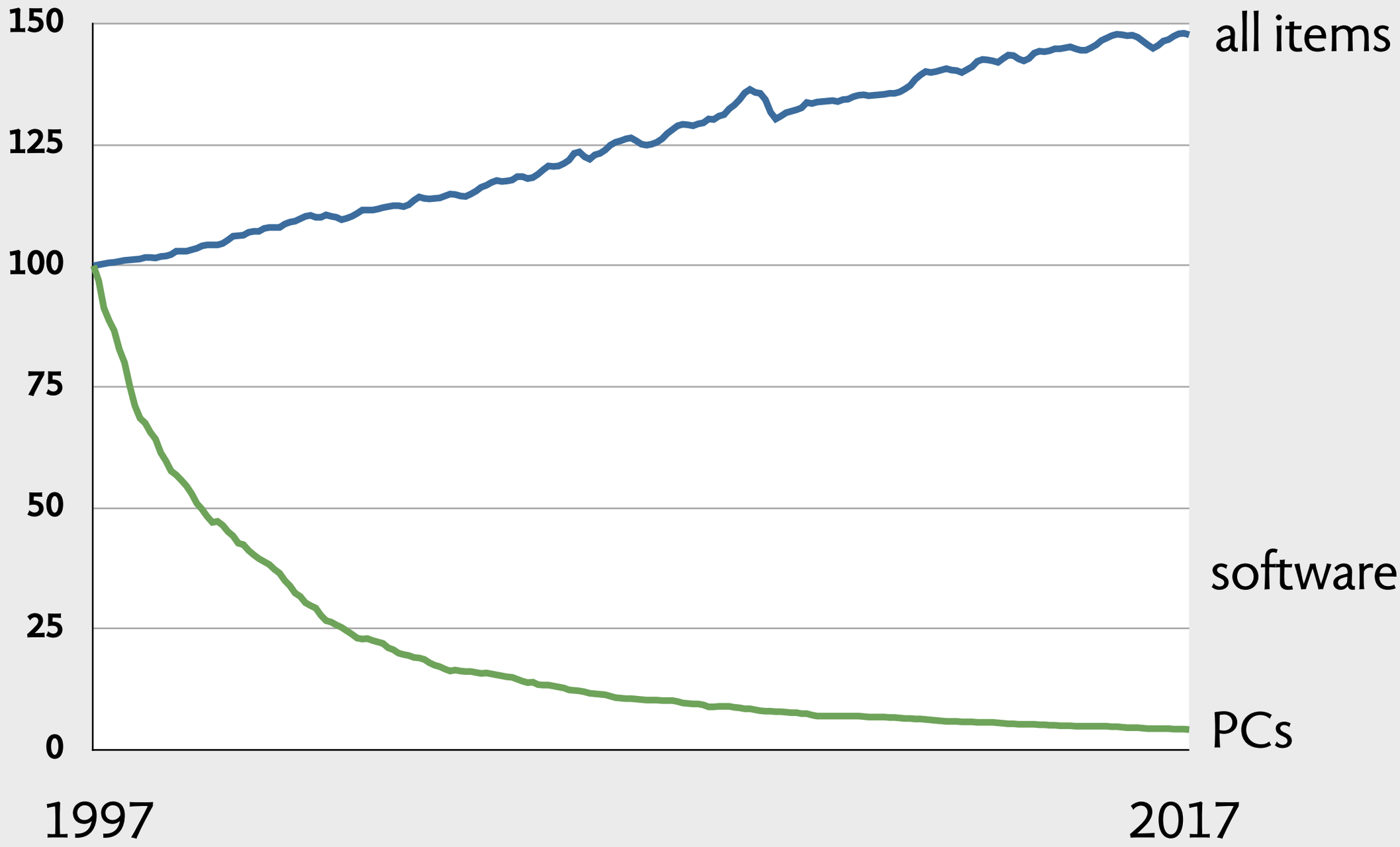
from bls.gov

consumer price index



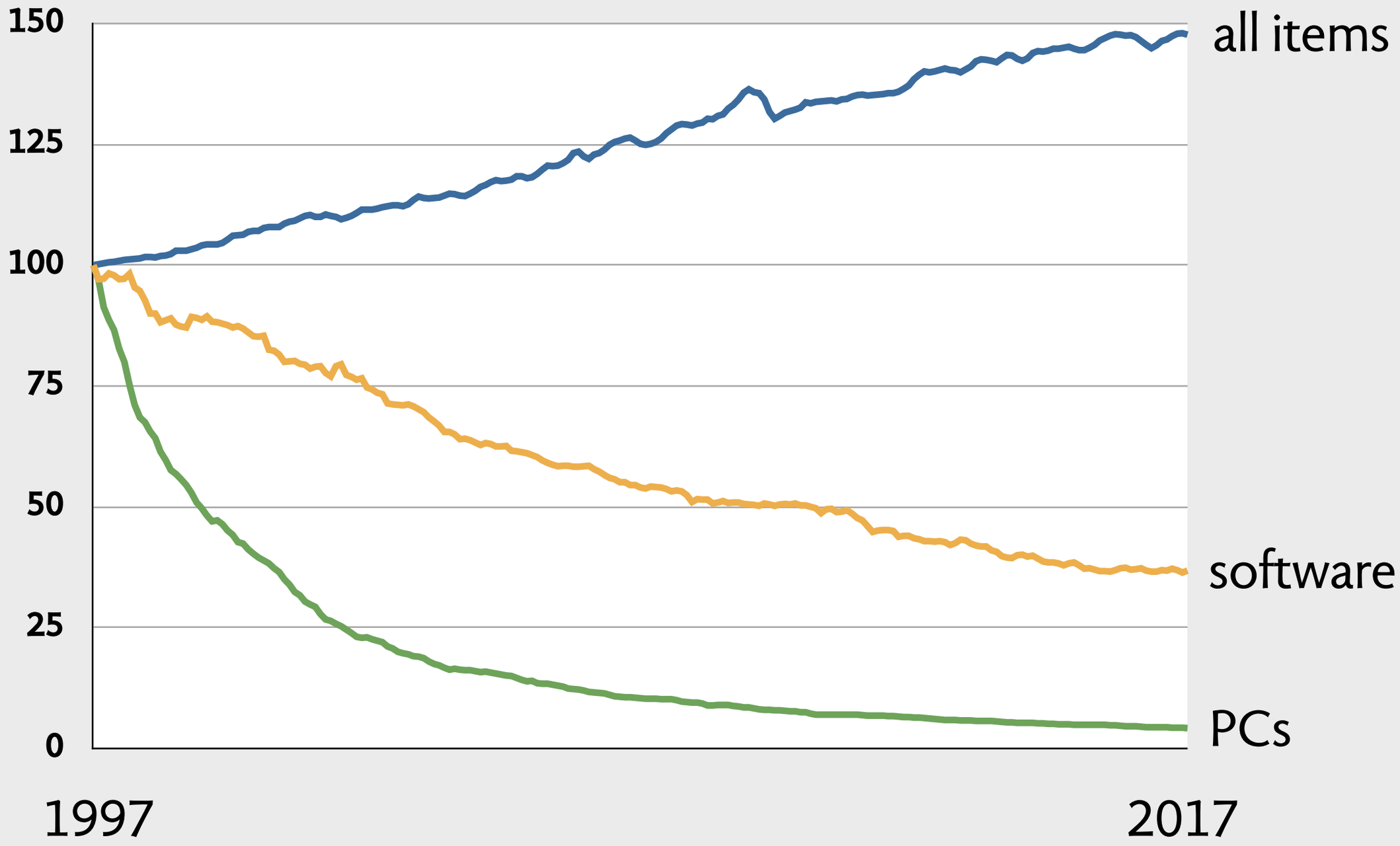
from bls.gov

consumer price index



from bls.gov

consumer price index



from bls.gov

costs of standard IT

costs of standard IT

buying off-the-shelf
may not fit your needs
paying for unused features

costs of standard IT

buying off-the-shelf
may not fit your needs
paying for unused features

hiring developers
means waiting, maybe years
costs $1-$100/line
unaffordable for small orgs

costs of standard IT

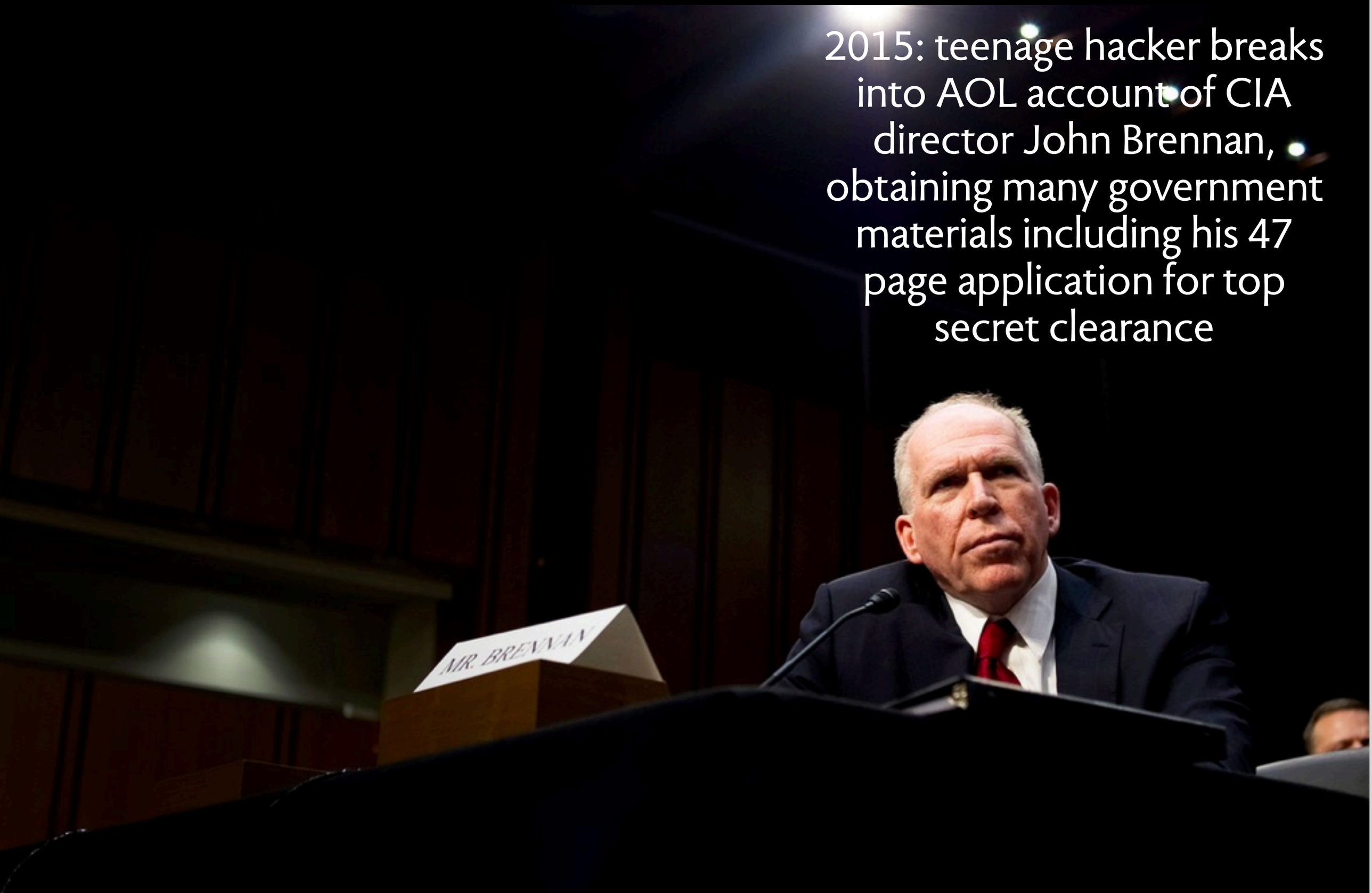
buying off-the-shelf
may not fit your needs
paying for unused features

hiring developers
means waiting, maybe years
costs $1-$100/line
unaffordable for small orgs

tweaking the code
is hard & dangerous
only by developers

costs of shadow IT

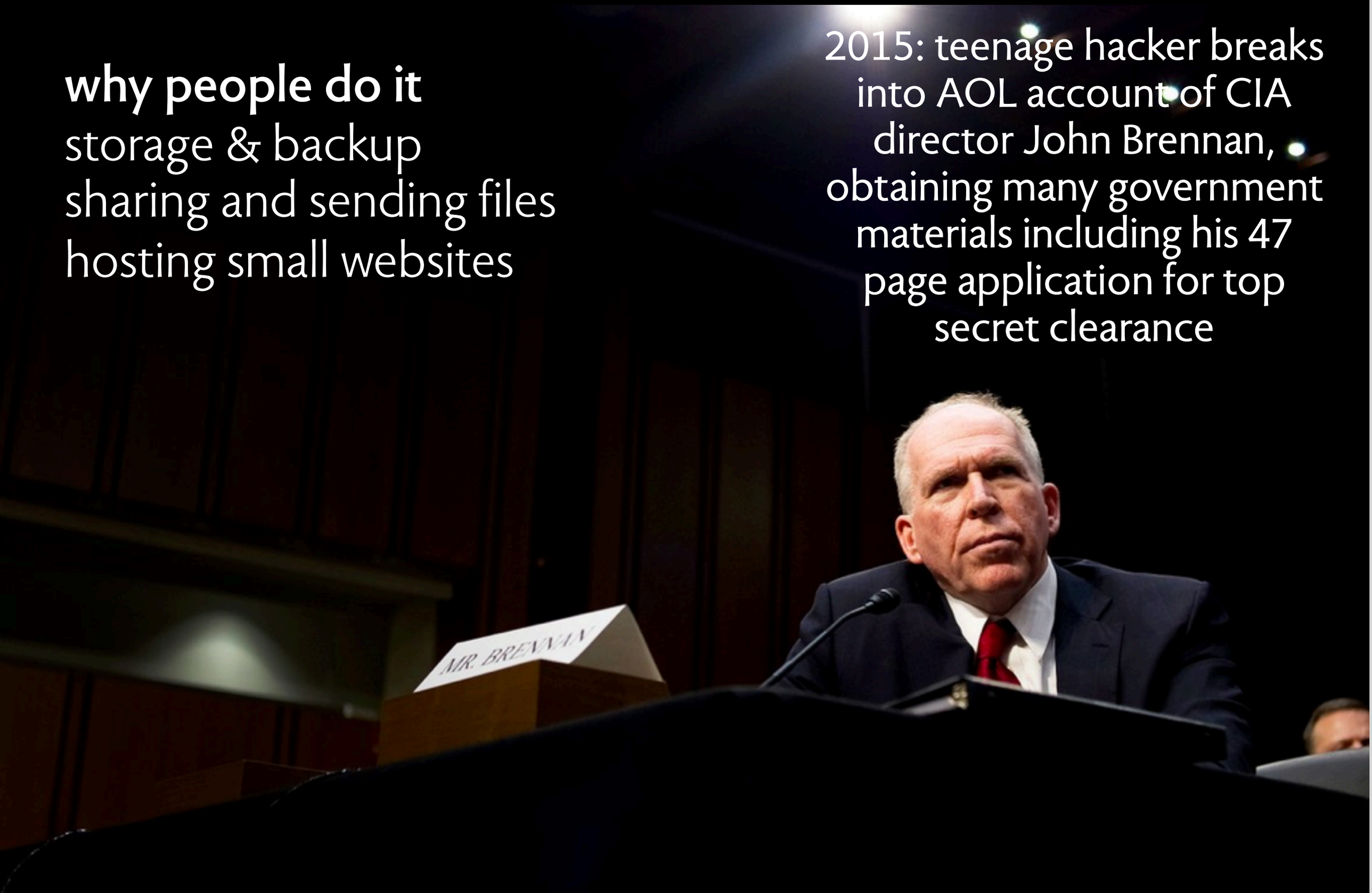
2015: teenage hacker breaks into AOL account of CIA director John Brennan, obtaining many government materials including his 47 page application for top secret clearance



costs of shadow IT

why people do it
storage & backup
sharing and sending files
hosting small websites

2015: teenage hacker breaks into AOL account of CIA director John Brennan, obtaining many government materials including his 47 page application for top secret clearance

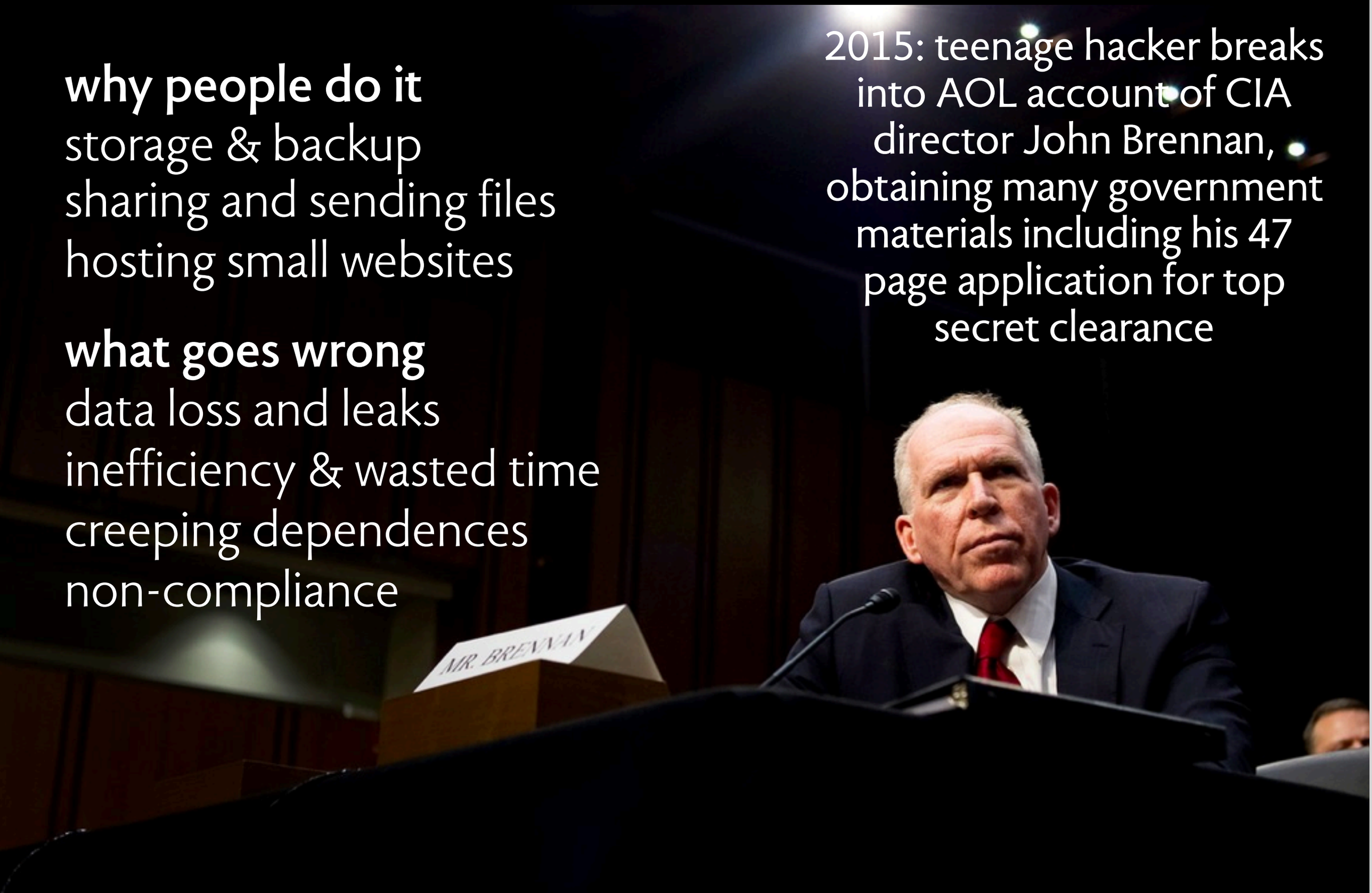


costs of shadow IT

why people do it
storage & backup
sharing and sending files
hosting small websites

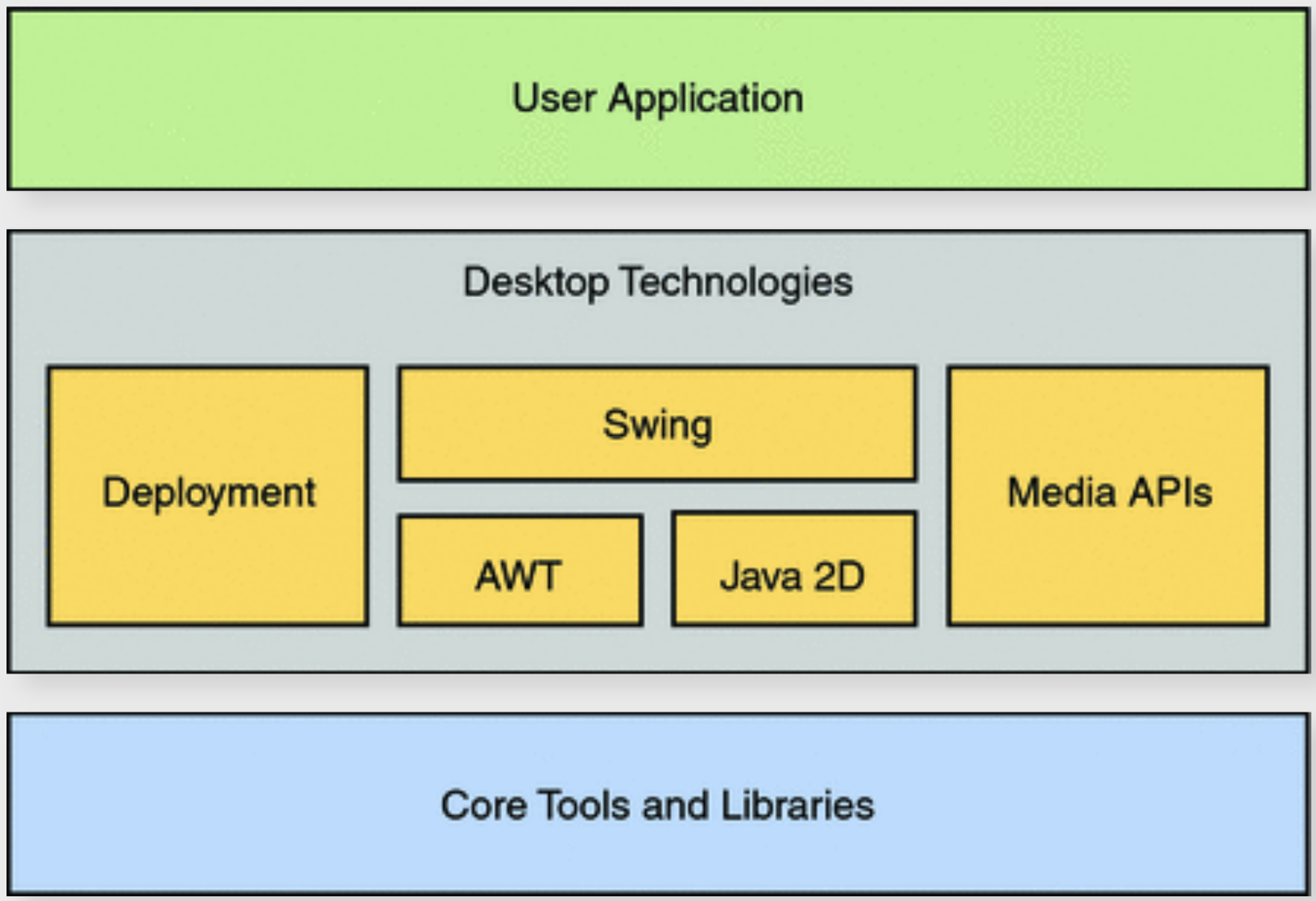
what goes wrong
data loss and leaks
inefficiency & wasted time
creeping dependences
non-compliance

2015: teenage hacker breaks into AOL account of CIA director John Brennan, obtaining many government materials including his 47 page application for top secret clearance



how we got here:
before the web

monolithic apps



Java SE Desktop

monolithic apps

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream(), true);
          out.println(new Date().toString());
        } finally {socket.close();}
      }
    }
    finally {listener.close();}
}}
```

using network to respond
to date requests

monolithic apps

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally {socket.close();}
      }
    }
    finally {listener.close();}
}}
```

using network to respond to date requests

using AWT to display a message

```
package awt;

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt applic
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent event) {
        System.exit(0);
      }
    });
  }
}
```

monolithic apps

uniform, explicit & simple dependences

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally {socket.close();}
      }
    }
    finally {listener.close();}
}}
```

using network to respond to date requests

using AWT to display a message

```
package awt;

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt applic
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent event) {
        System.exit(0);
      }
    });
  }
}
```

monolithic apps

uniform, explicit & simple dependences

statically typed interfaces

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally {socket.close();}
      }
    }
    finally {listener.close();}
}}
```

using network to respond to date requests

using AWT to display a message

```
package awt;

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt applic
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent event) {
        System.exit(0);
      }
    });
  }
}
```

monolithic apps

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally {socket.close();}
      }
    }
    finally {listener.close();}
}}
```

uniform, explicit & simple dependences

statically typed interfaces

using network to respond to date requests

using AWT to display a message

just one language

```
package awt;

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt applic
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent event) {
        System.exit(0);
      }
    });
  }
}
```

monolithic apps

uniform, explicit & simple dependences

statically typed interfaces

low level details

just one language

using network to respond to date requests

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally
      }
    }
    finally {l
}}
```

using AWT to display a message

```
package awt;

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt appli
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent event) {
        System.exit(0);
      }
    });
  }
}
```

monolithic apps

uniform, explicit & simple dependences

statically typed interfaces

no separation of concerns

low level details

just one language

using network to respond to date requests

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally
      }
    }
    finally {l
}}
```

using AWT to display a message

```
packag

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt appli
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new WindowAdapter() {
      public void windowClosing(WindowEvent event) {
        System.exit(0);
      }
    });
  }
}
```

monolithic apps

using network to respond to date requests

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally
      }
    }
    finally {l
}}
```

uniform, explicit & simple dependences

statically typed interfaces

no separation of concerns

low level details

just one language

using AWT to display a message

```
packag

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void main(String[] args) {
    Frame f=new Frame("Hello World example of awt applic
    Label label1=new Label("Hello World", Label.CENTER)
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new
      public void windowClo                    event) {
        System.exit(0);
      }
    });
  }
}
```

no separation of concerns

monolithic apps

using network to respond to date requests

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally
      }
    }
    finally {l
}}
```

uniform, explicit & simple dependences

statically typed interfaces

no separation of concerns

low level details

just one language

using AWT to display a message

```
packag

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void
    Frame f=new Frame
    Label label1=new
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new
      public void windowClo
        System.exit(0);
      }
    });
  }
}
```

spurious ordering

no separation of concerns

monolithic apps

using network to respond to date requests

```
import java.io.IOException;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.Date;

public class DateServer {

  public static void main(String[] args) throws IOException {
    ServerSocket listener = new ServerSocket(9090);
    try {
      while (true) {
        Socket socket = listener.accept();
        try {
          PrintWriter out =
            new PrintWriter(socket.getOutputStream
          out.println(new Date().toString());
        } finally
      }
    }
    finally {l
}}
```

uniform, explicit & simple dependences

statically typed interfaces

verbose language

no separation of concerns

low level details

just one language

using AWT to display a message

```
packag

import java.awt.Frame;
import java.awt.Label;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

public class Hello {

  public static void
    Frame f=new Frame
    Label label1=new
    f.add(label1);
    f.setSize(300,100);
    f.setVisible(true);
    f.addWindowListener(new
      public void windowClo
        System.exit(0);
      }
    });
  }
}
```

spurious ordering

no separation of concerns

what the web wanted

what the web wanted

a wish list
manipulate small data
use a database backend
interact with a client UI
separate concerns

what the web wanted

a wish list
manipulate small data
use a database backend
interact with a client UI
separate concerns

but none is easy in Java...

the web arrives

manipulate small data

"Java is to JavaScript as ham is to hamster" *Jeremy Keith*

manipulate small data

"Java is to JavaScript as ham is to hamster" *Jeremy Keith*

example: flattening a list (from rosettacode.org)

from: [[1], 2, [[3, 4], 5], [[[]]], [[[6]]], 7, 8, []]
to: [1, 2, 3, 4, 5, 6, 7, 8]

manipulate small data

"Java is to JavaScript as ham is to hamster" *Jeremy Keith*

example: flattening a list (from rosettacode.org)

from: [[1], 2, [[3, 4], 5], [[[]]], [[[6]]], 7, 8, []]
to: [1, 2, 3, 4, 5, 6, 7, 8]

Java

```java
import java.util.LinkedList;
import java.util.List;

public final class FlattenUtil {

    public static List<Object> flatten(List<?> list) {
        List<Object> retVal = new LinkedList<Object>();
        flatten(list, retVal);
        return retVal;
    }

    public static void flatten(List<?> fromTreeList, List<Object> toFlatList) {
        for (Object item : fromTreeList) {
            if (item instanceof List<?>) {
                flatten((List<?>) item, toFlatList);
            } else {
                toFlatList.add(item);
            }
        }
    }
}
```

manipulate small data

"Java is to JavaScript as ham is to hamster" *Jeremy Keith*

example: flattening a list (from rosettacode.org)

from: [[1], 2, [[3, 4], 5], [[[]]], [[[6]]], 7, 8, []]
to: [1, 2, 3, 4, 5, 6, 7, 8]

Java

```java
import java.util.LinkedList;
import java.util.List;

public final class FlattenUtil {

    public static List<Object> flatten(List<?> list) {
        List<Object> retVal = new LinkedList<Object>();
        flatten(list, retVal);
        return retVal;
    }

    public static void flatten(List<?> fromTreeList, List<Object> toFlatList) {
        for (Object item : fromTreeList) {
            if (item instanceof List<?>) {
                flatten((List<?>) item, toFlatList);
            } else {
                toFlatList.add(item);
            }
        }
    }
}
```

Javascript

```javascript
function flatten(list) {
  return list.reduce(function (acc, val) {
    return acc.concat(val.constructor === Array ? flatten(val) : val);
  }, []);
}
```

manipulate small data

"Java is to JavaScript as ham is to hamster" *Jeremy Keith*

example: flattening a list (from rosettacode.org)

from: [[1], 2, [[3, 4], 5], [[[]]], [[[6]]], 7, 8, []]
to: [1, 2, 3, 4, 5, 6, 7, 8]

Java

```java
import java.util.LinkedList;
import java.util.List;

public final class FlattenUtil {

    public static List<Object> flatten(List<?> list) {
        List<Object> retVal = new LinkedList<Object>();
        flatten(list, retVal);
        return retVal;
    }

    public static void flatten(List<?> fromTreeList, List<Object> toFlatList) {
        for (Object item : fromTreeList) {
            if (item instanceof List<?>) {
                flatten((List<?>) item, toFlatList);
            } else {
                toFlatList.add(item);
            }
        }
    }
}
```

no need to figure out types

Javascript

```javascript
function flatten(list) {
  return list.reduce(function (acc, val) {
    return acc.concat(val.constructor === Array ? flatten(val) : val);
  }, []);
}
```

use a database backend

```
sql = "Select * from Users where" +
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'"

user_array = ActiveRecord::Base.connection.execute(sql)
```

Rails raw SQL query

use a database backend

```
sql = "Select * from Users where" +
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'"

user_array = ActiveRecord::Base.connection.execute(sql)
```

easy construction of query as a string

Rails raw SQL query

separate concerns

```
var counter = 0;

app.get('/show', function (req, res) {
  res.send('Counter value is ' + counter);
});

app.get('/reset', function (req, res) {
  counter = 0;
  res.send('Counter value reset');
});

app.get('/inc', function (req, res) {
  counter++;
  res.send('Counter value incremented');
});
```

separate concerns

route separated from function

```
var counter = 0;

app.get('/show', function (req, res) {
  res.send('Counter value is ' + counter);
});

app.get('/reset', function (req, res) {
  counter = 0;
  res.send('Counter value reset');
});

app.get('/inc', function (req, res) {
  counter++;
  res.send('Counter value incremented');
});
```

separate concerns

route separated from function

```
var counter = 0;

app.get('/show', function (req, res) {
  res.send('Counter value is ' + counter);
});

app.get('/reset', function (req, res) {
  counter = 0;
  res.send('Counter value reset');
});

app.get('/inc', function (req, res) {
  counter++;
  res.send('Counter value incremented');
});
```

```
var users = require('./routes/users');
app.use('/users', users);
```

separate concerns

route separated from function

```
var counter = 0;

app.get('/show', function (req, res) {
  res.send('Counter value is ' + counter);
});

app.get('/reset', function (req, res) {
  counter = 0;
  res.send('Counter value reset');
});

app.get('/inc', function (req, res) {
  counter++;
  res.send('Counter value incremented');
});
```

```
var users = require('./routes/users');
app.use('/users', users);
```

handlers for routes in separate files

separate concerns

route separated from function

```
var counter = 0;

app.get('/show', function (req, res) {
 res.send('Counter value is ' + counter);
});

app.get('/reset', function (req, res) {
 counter = 0;
 res.send('Counter value reset');
});

app.get('/inc', function (req, res) {
 counter++;
 res.send('Counter value incremented');
});
```

```
var users = require('./routes/users');
app.use('/users', users);
```

handlers for routes in separate files

```
<html><body>
Counter value is {{counter}}
<form action="/inc" method="post">
 <input type="submit" value="inc by">
 <input type="text" name="by"
value="1">
</form>
</body></html>
```

```
var counter = 0;

app.get('/', function (req, res) {
 res.render('index', {counter: counter});
});
```

separate concerns

route separated from function

templates separate view from control

```
var counter = 0;

app.get('/show', function (req, res) {
 res.send('Counter value is ' + counter);
});

app.get('/reset', function (req, res) {
 counter = 0;
 res.send('Counter value reset');
});

app.get('/inc', function (req, res) {
 counter++;
 res.send('Counter value incremented');
});
```

```
var users = require('./routes/users');
app.use('/users', users);
```

handlers for routes in separate files

```
<html><body>
Counter value is {{counter}}
<form action="/inc" method="post">
 <input type="submit" value="inc by">
 <input type="text" name="by"
value="1">
</form>
</body></html>
```

```
var counter = 0;

app.get('/', function (req, res) {
 res.render('index', {counter: counter});
});
```

interact with a client UI

```
app.use(session({ secret : 'foo', resave : true,
saveUninitialized : true }));

app.get('/:name', function (req, res) {
  req.session.name = req.params.name;
  res.send('Hello ' + req.params.name);
});

app.get('/', function (req, res) {
  res.send('Welcome back ' + req.session.name);
});
```

a server that remembers your name

interact with a client UI

save name in session

```
app.use(session({ secret : 'foo', resave : true,
saveUninitialized : true }));

app.get('/:name', function (req, res) {
  req.session.name = req.params.name;
  res.send('Hello ' + req.params.name);
});

app.get('/', function (req, res) {
  res.send('Welcome back ' + req.session.name);
});
```

a server that remembers your name

interact with a client UI

save name in session

```
app.use(session({ secret : 'foo', resave : true,
saveUninitialized : true }));

app.get('/:name', function (req, res) {
  req.session.name = req.params.name;
  res.send('Hello ' + req.params.name);
});

app.get('/', function (req, res) {
  res.send('Welcome back ' + req.session.name);
});
```

return name from session

a server that remembers your name

interact with a client UI

save name in session

```
app.use(session({ secret : 'foo', resave : true,
saveUninitialized : true }));

app.get('/:name', function (req, res) {
 req.session.name = req.params.name;
 res.send('Hello ' + req.params.name);
});

app.get('/', function (req, res) {
 res.send('Welcome back ' + req.session.name);
});
```

session structure hides cookies + db

return name from session

a server that remembers your name

so are we happy now?

JavaScript

"the duct tape of the Internet"

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"
> 1 / "hello"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"
> 1 / "hello"
NaN
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"

> 1 / "hello"
NaN

> x = 1 / "hello"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"

> 1 / "hello"
NaN

> x = 1 / "hello"

> (x == NaN) ? "bad": "good"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"

> 1 / "hello"
NaN

> x = 1 / "hello"

> (x == NaN) ? "bad": "good"
"good"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"

> 1 / "hello"
NaN

> x = 1 / "hello"

> (x == NaN) ? "bad": "good"
"good"

> typeof(NaN)
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"

> 1 / "hello"
NaN

> x = 1 / "hello"

> (x == NaN) ? "bad": "good"
"good"

> typeof(NaN)
"number"
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"
> 1 / "hello"
NaN
> x = 1 / "hello"
> (x == NaN) ? "bad": "good"
"good"
> typeof(NaN)
"number"
> NaN === NaN
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"
> 1 / "hello"
NaN
> x = 1 / "hello"
> (x == NaN) ? "bad": "good"
"good"
> typeof(NaN)
"number"
> NaN === NaN
false
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"
> 1 / "hello"
NaN
> x = 1 / "hello"
> (x == NaN) ? "bad": "good"
"good"
> typeof(NaN)
"number"
> NaN === NaN
false
> NaN !== NaN
```

JavaScript

"the duct tape of the Internet"

```
> 1 + "hello"
"1hello"

> 1 / "hello"
NaN

> x = 1 / "hello"

> (x == NaN) ? "bad": "good"
"good"

> typeof(NaN)
"number"

> NaN === NaN
false

> NaN !== NaN
true
```

coming of age

If Javascript were a person, what life event would it be going through this year?

a. Getting a driver's license (16)

b. Voting in its first election (18)

c. Buying its first legal drink (21)

d. Getting a discount on car insurance (25)

coming of age

If Javascript were a person, what life event would it be going through this year?

a. Getting a driver's license (16)

b. Voting in its first election (18)

c. Buying its first legal drink (21)

d. Getting a discount on car insurance (25)

and the answer is...

JS

coming of age

If Javascript were a person, what life event would it be going through this year?

a. Getting a driver's license (16)

b. Voting in its first election (18)

c. Buying its first legal drink (21)

d. Getting a discount on car insurance (25)

and the answer is...

JS

database complexities

embedded SQL: not a good match for objects

```
sql = "Select * from Users where" +
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'"

user_array = ActiveRecord::Base.connection.execute(sql)
```

database complexities

embedded SQL: not a good match for objects

```
sql = "Select * from Users where" +
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'"

user_array = ActiveRecord::Base.connection.execute(sql)
```

using an “object relational mapper”

```
user = User.where(
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'")
```

database complexities

embedded SQL: not a good match for objects

```
sql = "Select * from Users where" +
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'"

user_array = ActiveRecord::Base.connection.execute(sql)
```

using an “object relational mapper”

```
user = User.where(
    "name = '#{params[:name]}'" +
    "AND password = '#{params[:password]}'")
```

(and both still have an injection vulnerability)

endless frameworks

"No JavaScript frameworks were created during the writing of this article"

from: *How it feels to learn JavaScript in 2016*

callback hell

what we used to write

```
do_a();
do_b();
do_c();
```

callback hell

what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

callback hell

what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

what inspired this madness?

callback hell

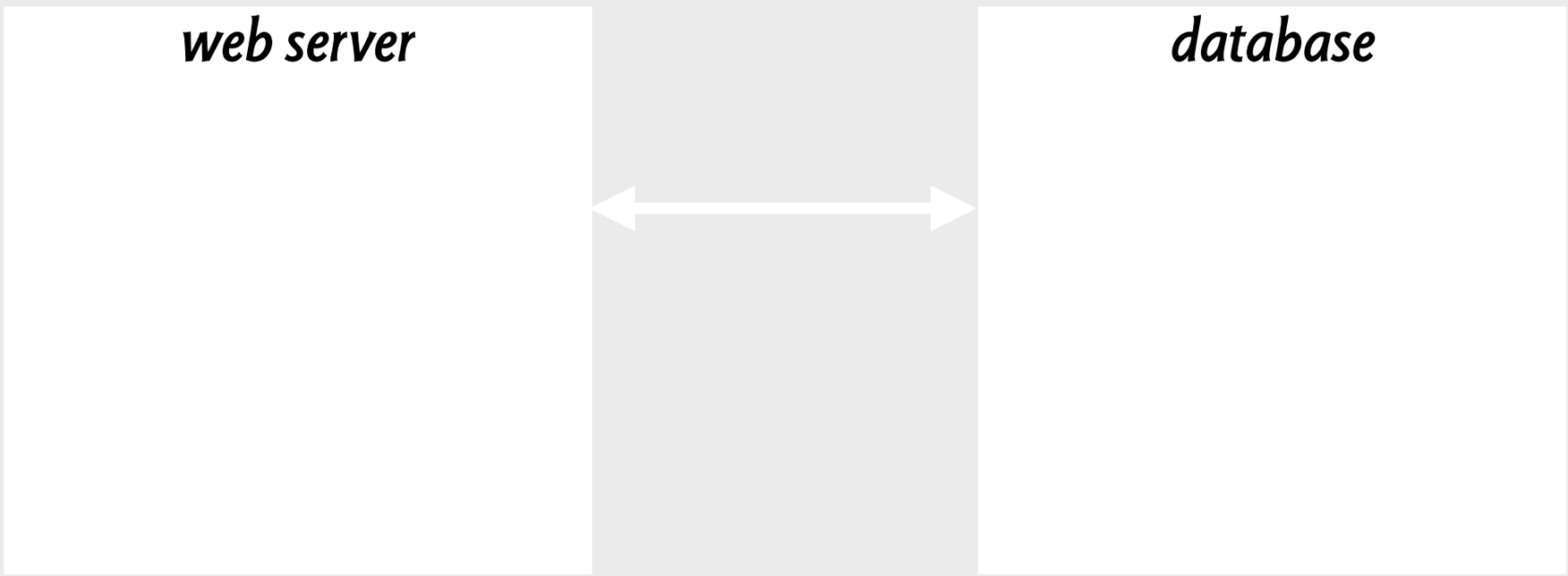
what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

what inspired this madness?



callback hell

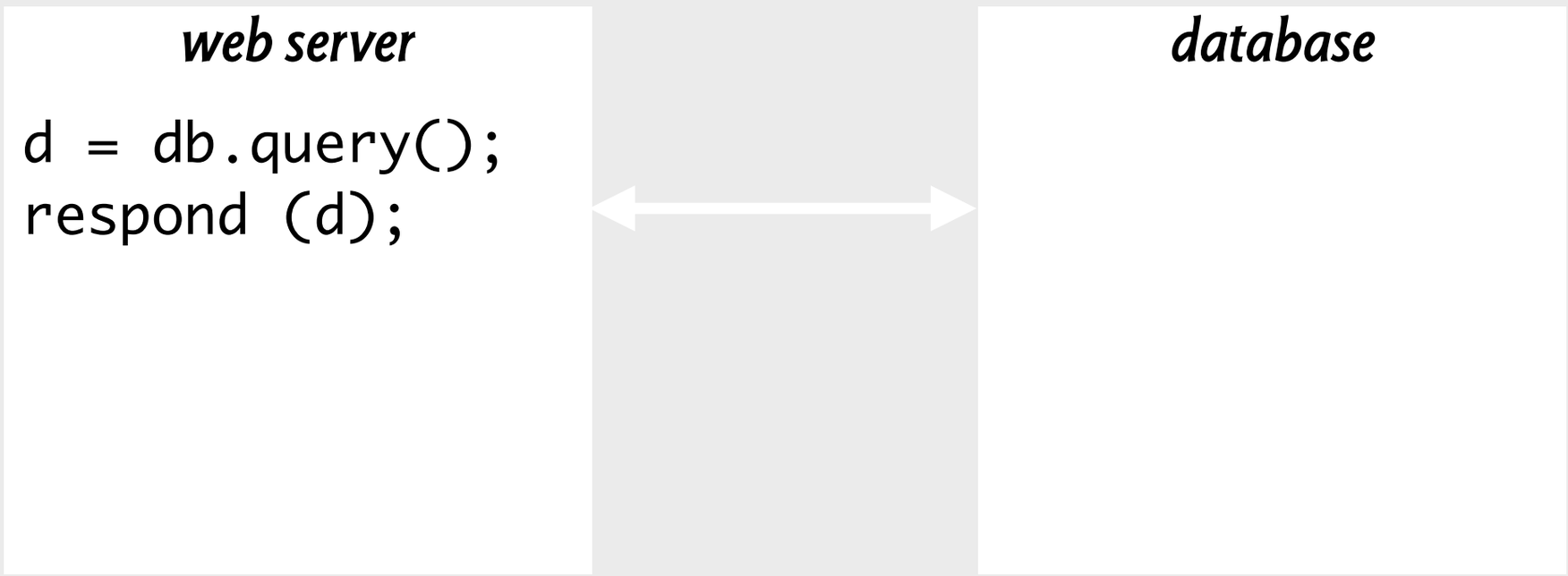
what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

what inspired this madness?



callback hell

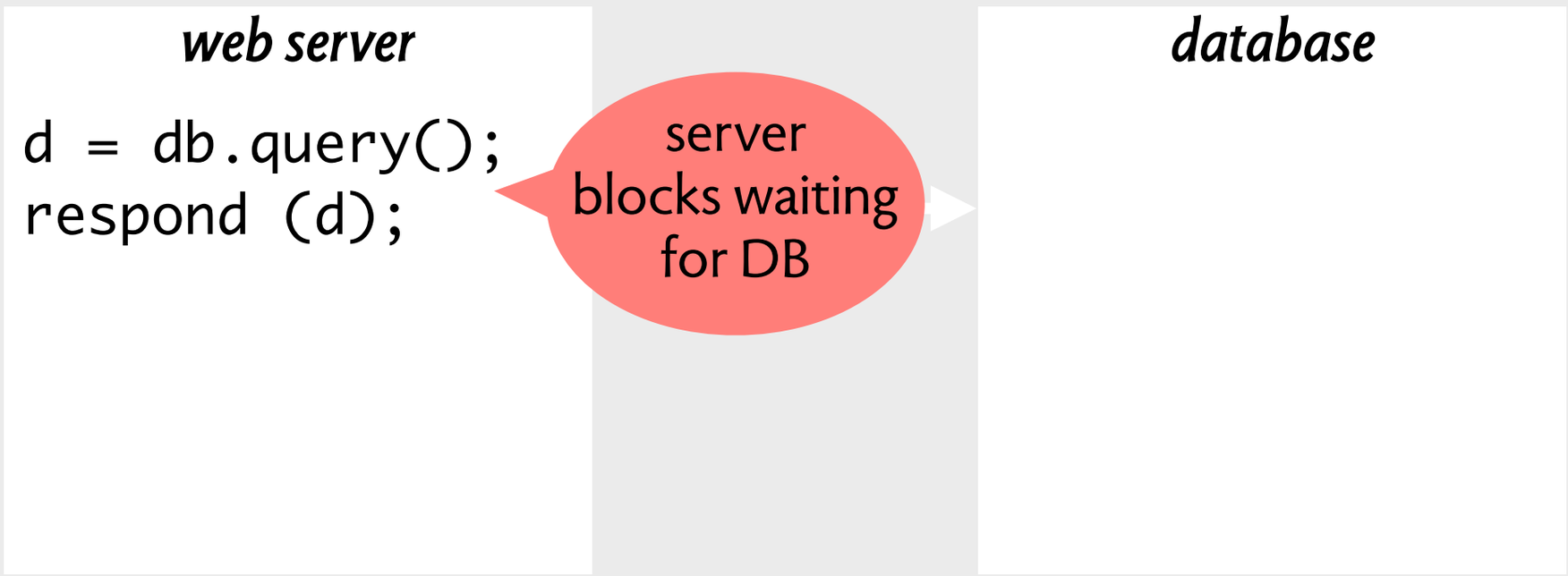
what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

what inspired this madness?



callback hell

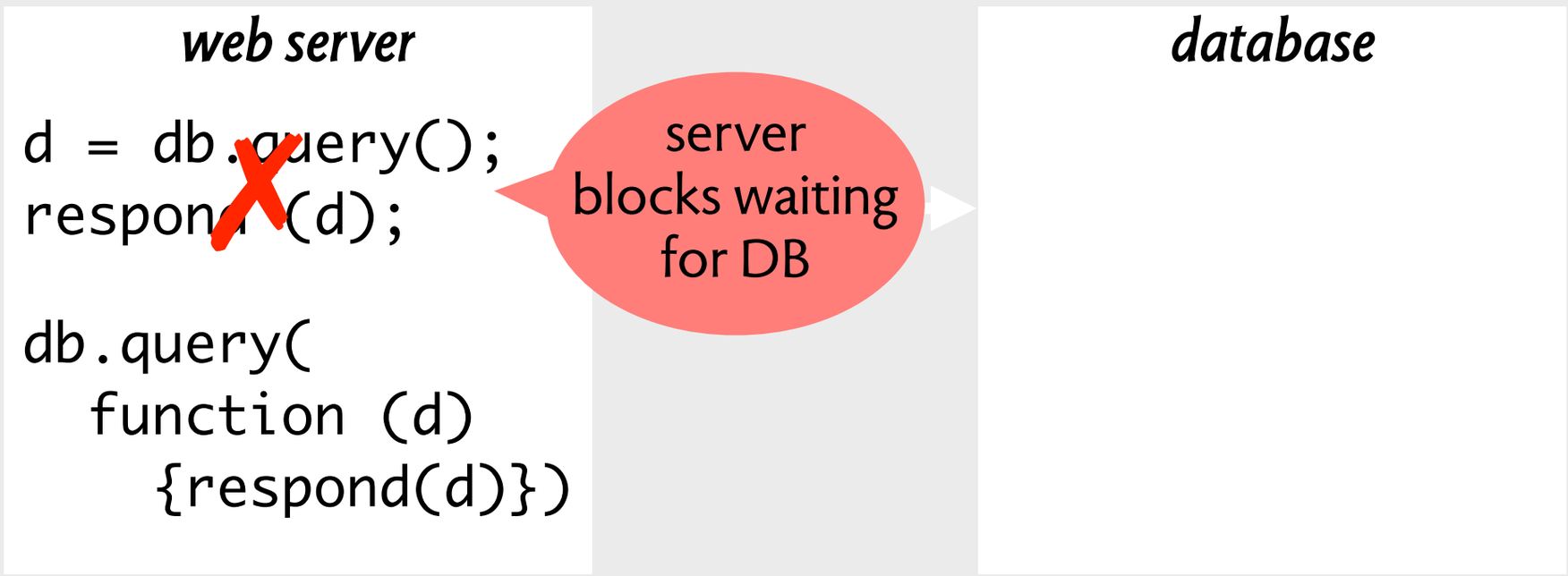
what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

what inspired this madness?



callback hell

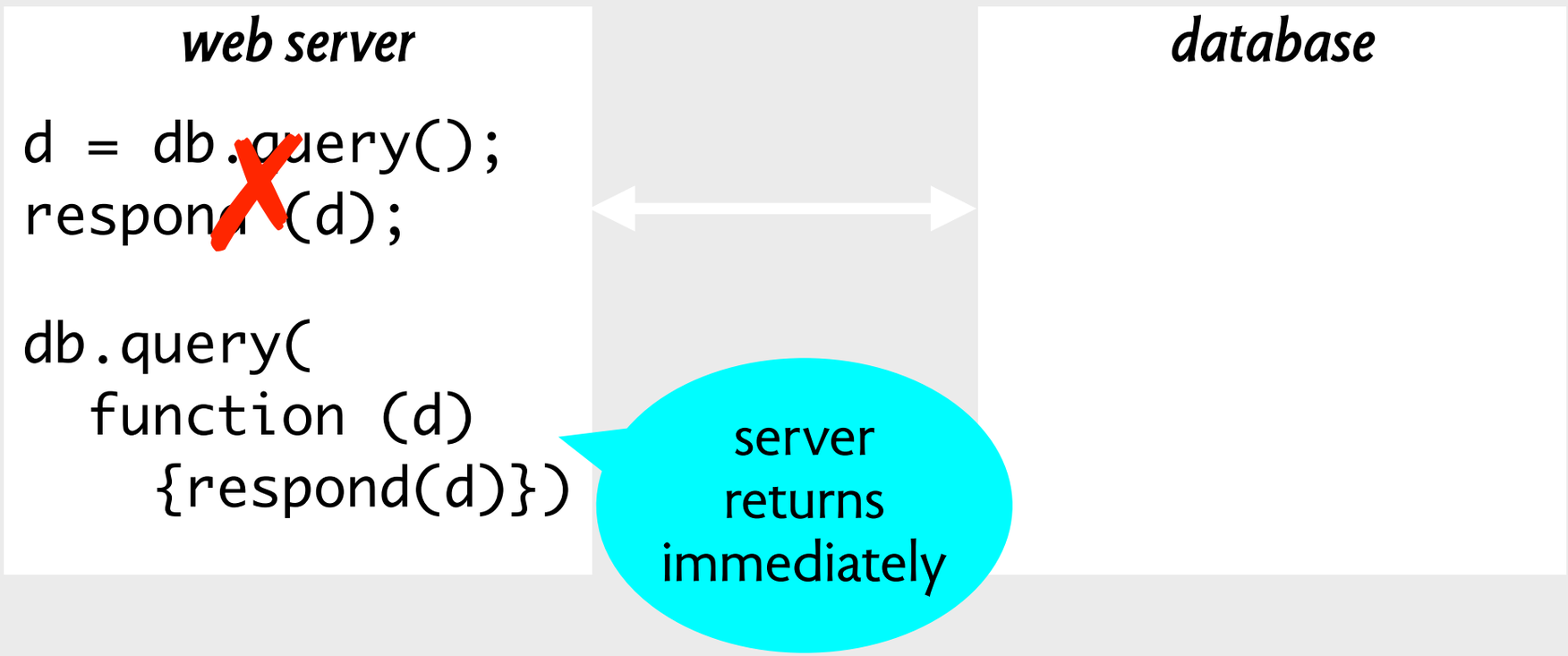
what we used to write

```
do_a();
do_b();
do_c();
```

what we write now

```
do_a (function () {
   do_b (function () {
      do_c(...)
   })
})
```

what inspired this madness?



“low code”

to the rescue

been there, done that?

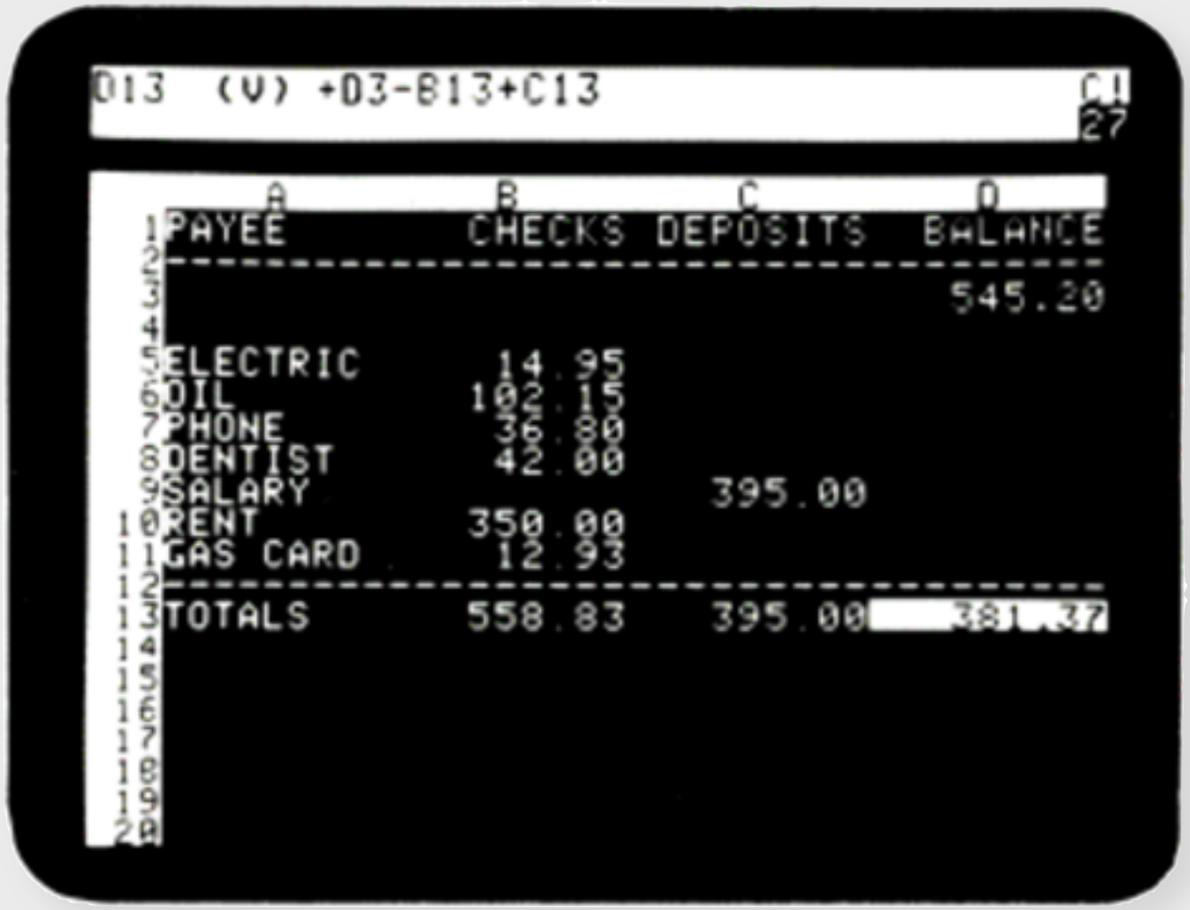
Though hard to describe in words, ??? comes alive visually. In minutes, people who have never used a computer are writing and using programs. Although you are operating in plain English, the program is being executed in machine language. But as far as you're concerned, the entire procedure is software transparent. You simply write on this so-called electronic blackboard what you would like it to do -- and it does it.

Ben Rosen, Morgan Stanley Electronics Letter (1979)

been there, done that?

Though hard to describe in words, Visicalc comes alive visually. In minutes, people who have never used a computer are writing and using programs. Although you are operating in plain English, the program is being executed in machine language. But as far as you're concerned, the entire procedure is software transparent. You simply write on this so-called electronic blackboard what you would like it to do -- and it does it.

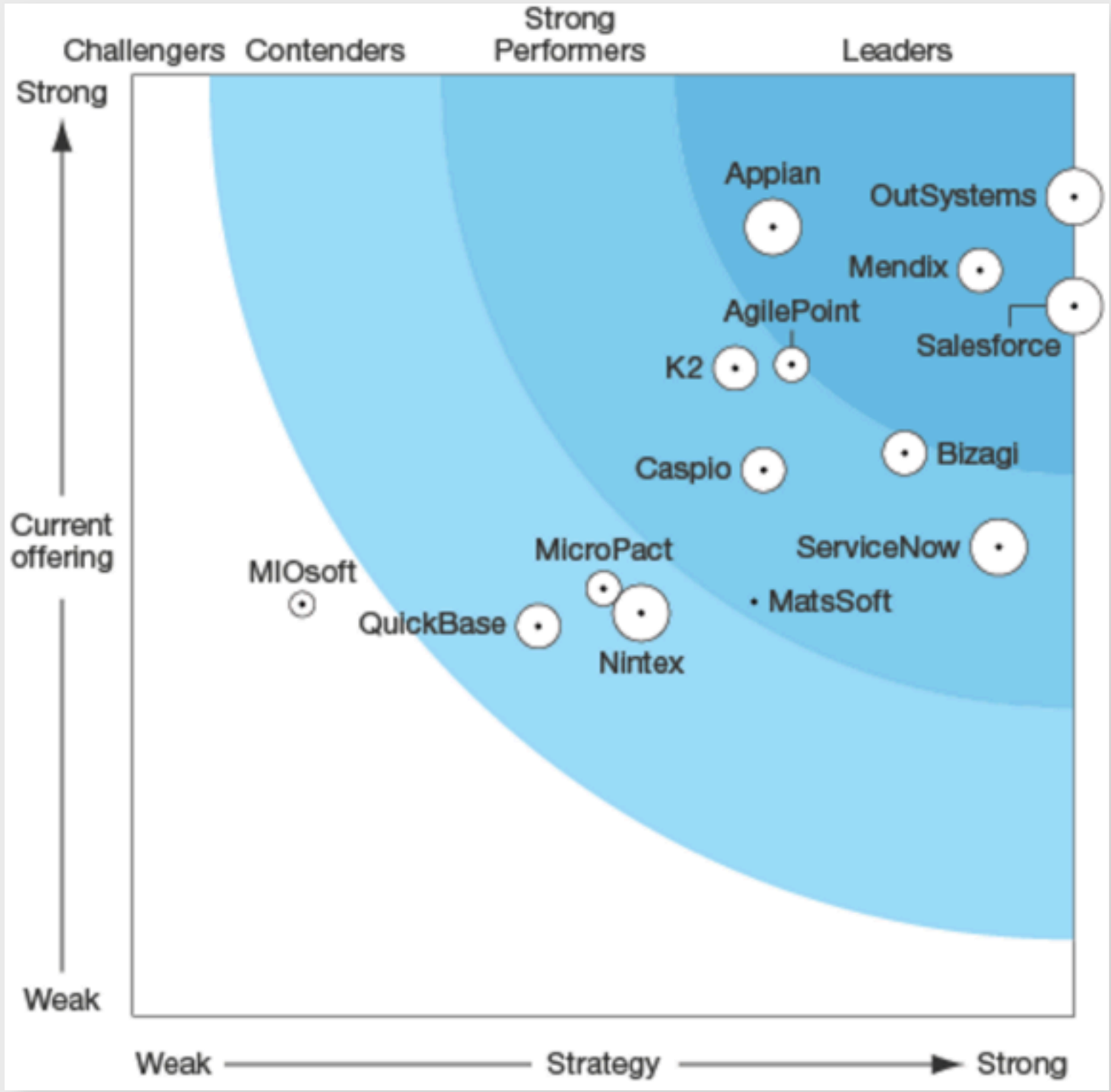
Ben Rosen, Morgan Stanley Electronics Letter (1979)



"low code" platforms

term coined by Forrester Research in 2014

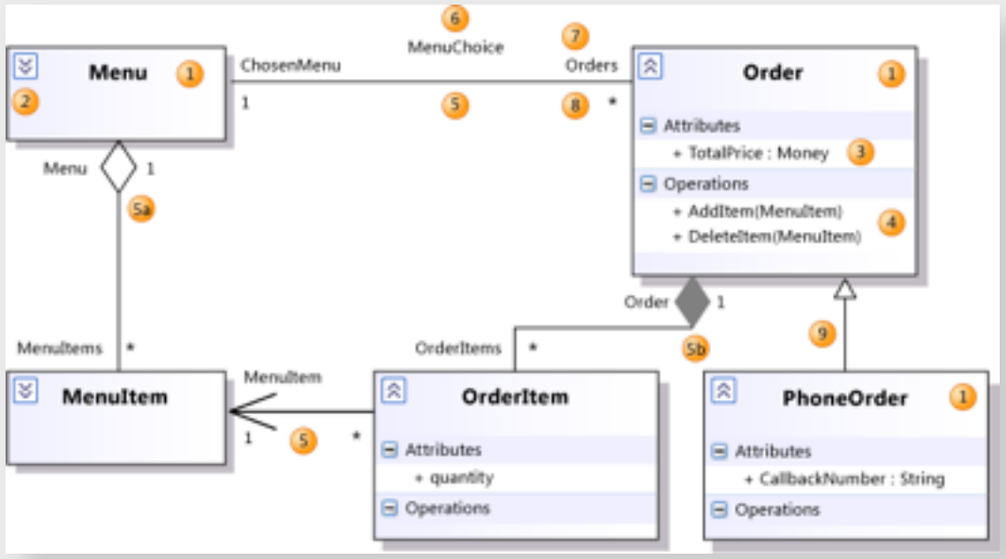
Forrester: market for low-code dev will be $15.5B by 2020



technology origins

technology origins

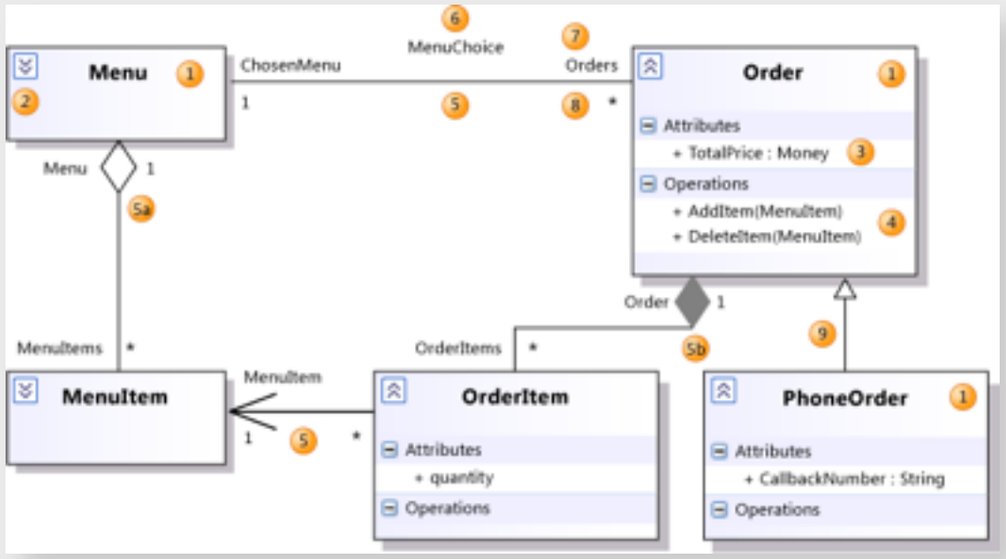
model-driven development



UML class diagram, Visual Studio (OMT, 1991)

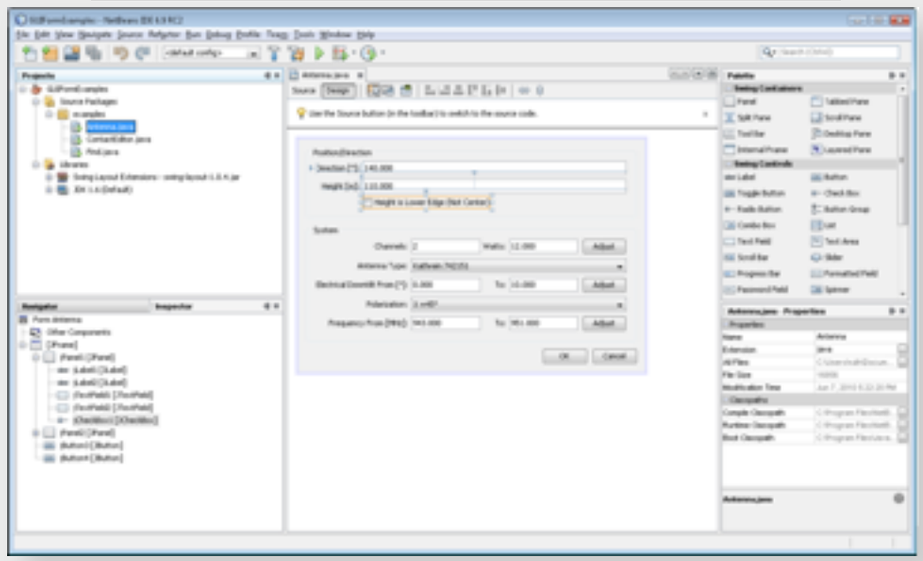
technology origins

model-driven development



UML class diagram, Visual Studio (OMT, 1991)

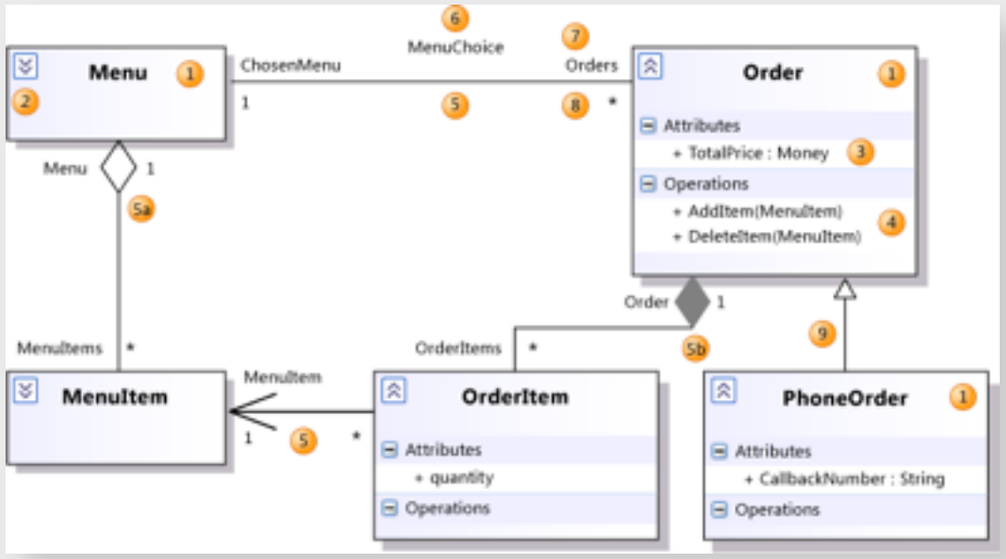
user interface builders



Netbeans GUI Builder (Project Matisse, 2008)

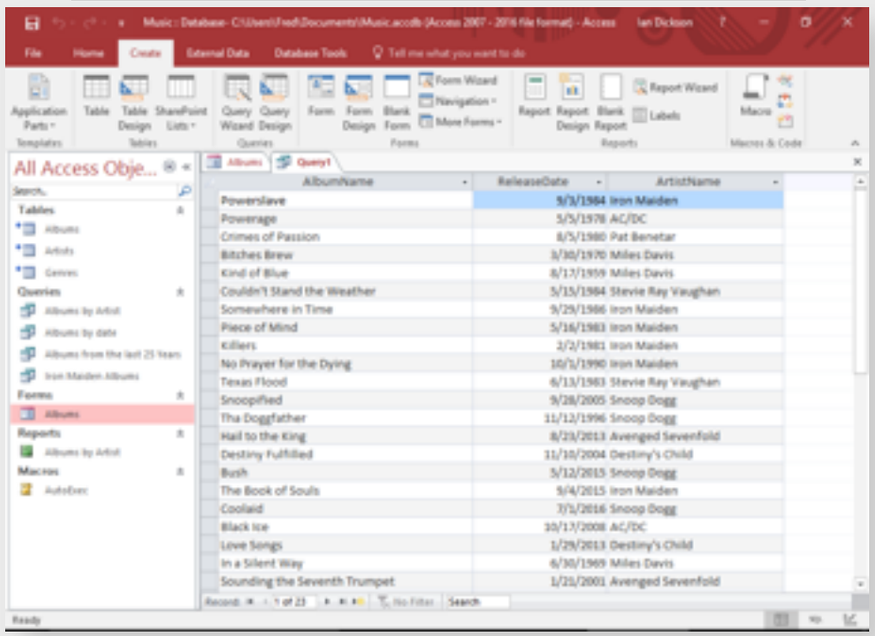
technology origins

model-driven development



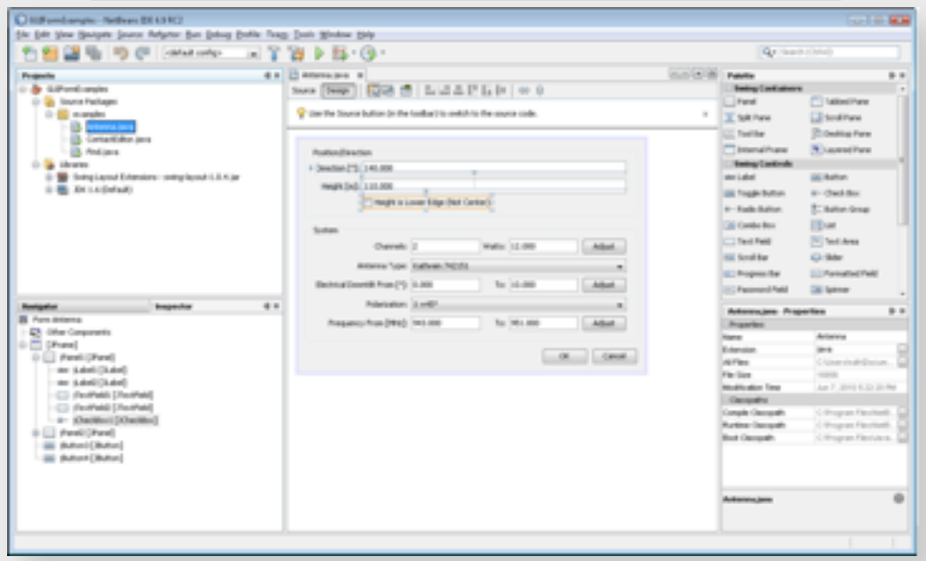
UML class diagram, Visual Studio (OMT, 1991)

groupware databases



Microsoft Access (first release 1992)

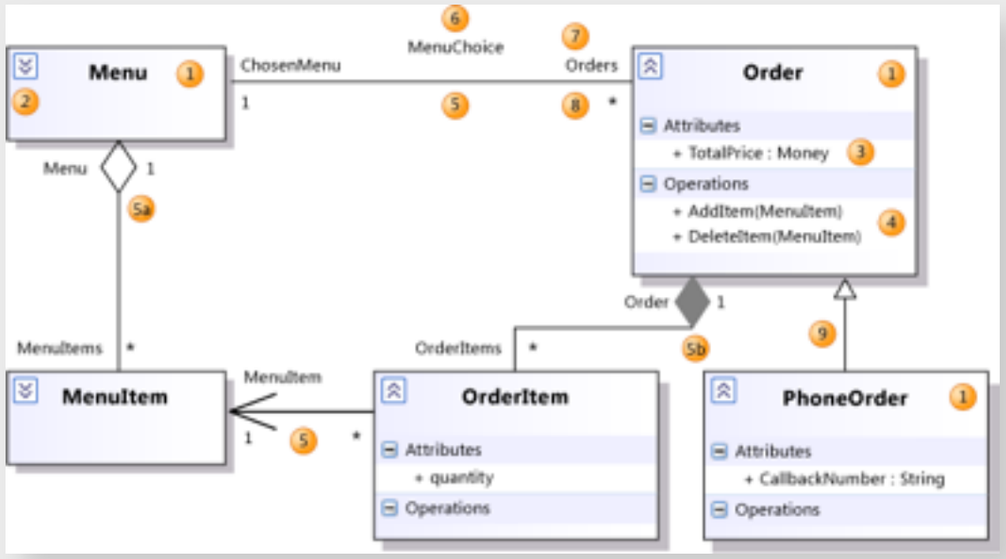
user interface builders



Netbeans GUI Builder (Project Matisse, 2008)

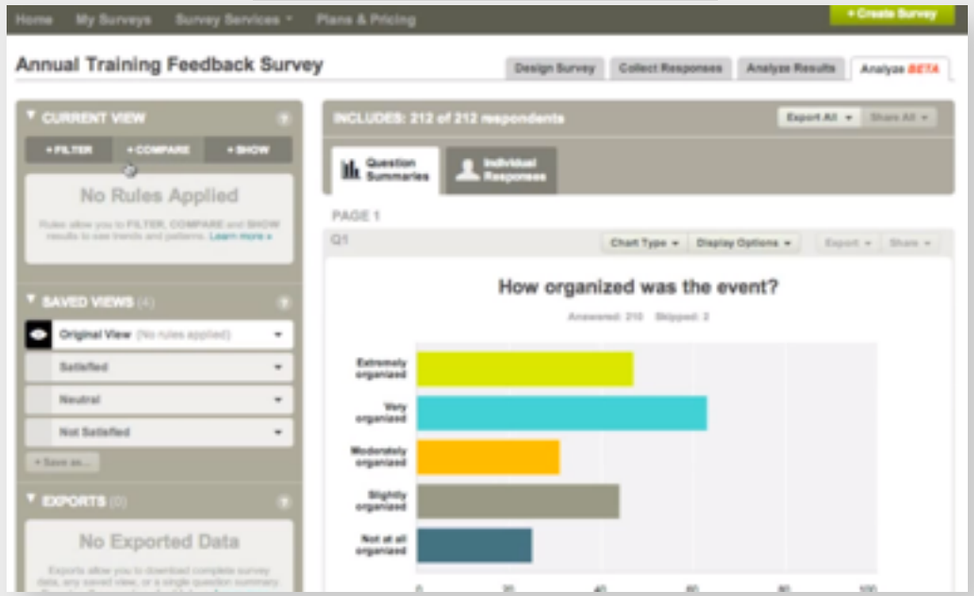
technology origins

model-driven development



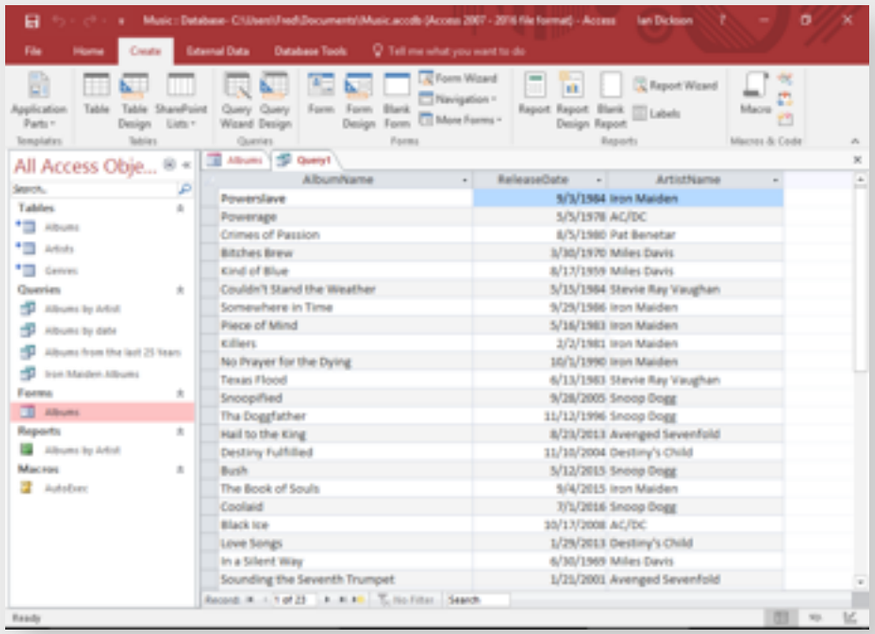
UML class diagram, Visual Studio (OMT, 1991)

form builders



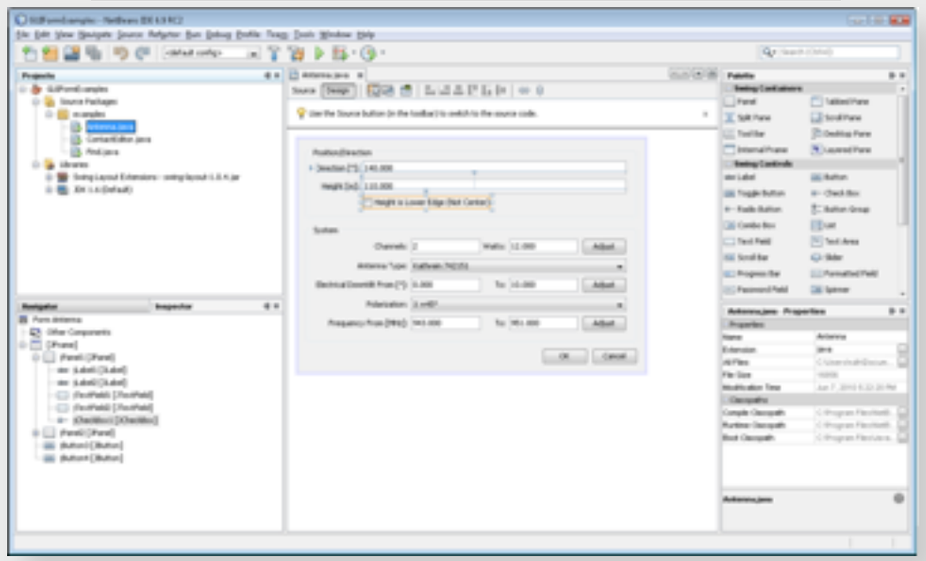
Survey Monkey (founded 1999)

groupware databases



Microsoft Access (first release 1992)

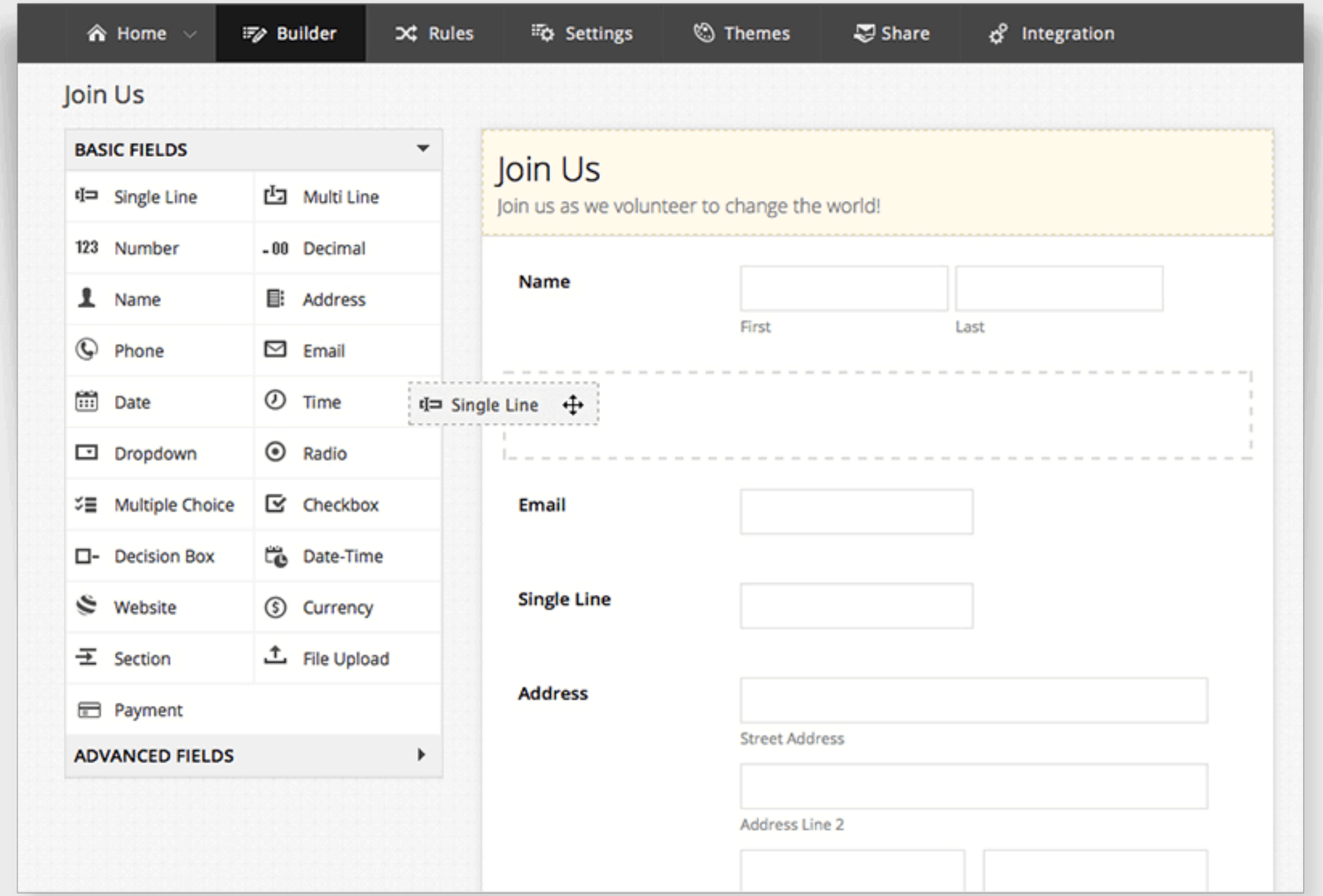
user interface builders



Netbeans GUI Builder (Project Matisse, 2008)

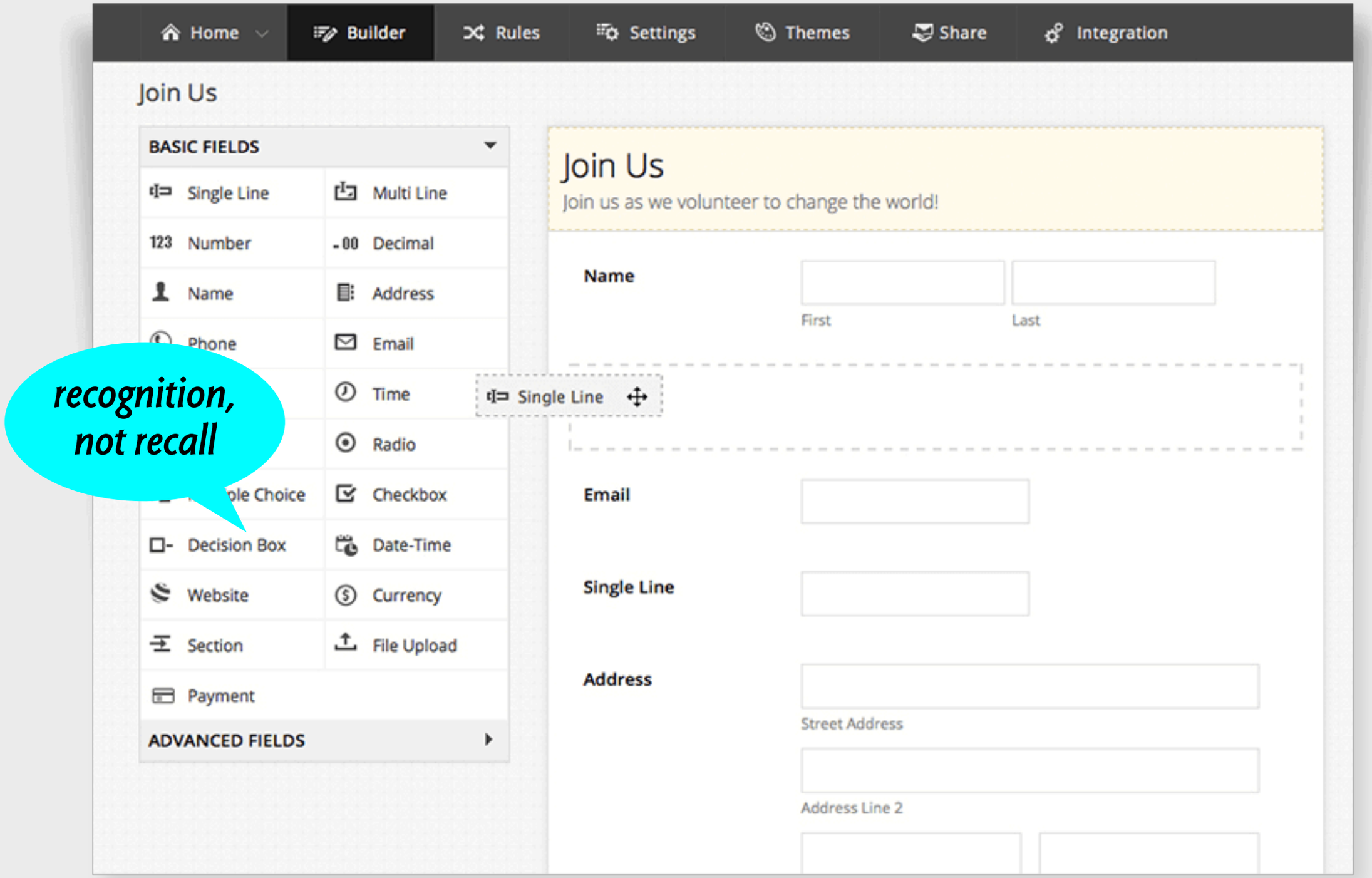
ingredients

visual editing



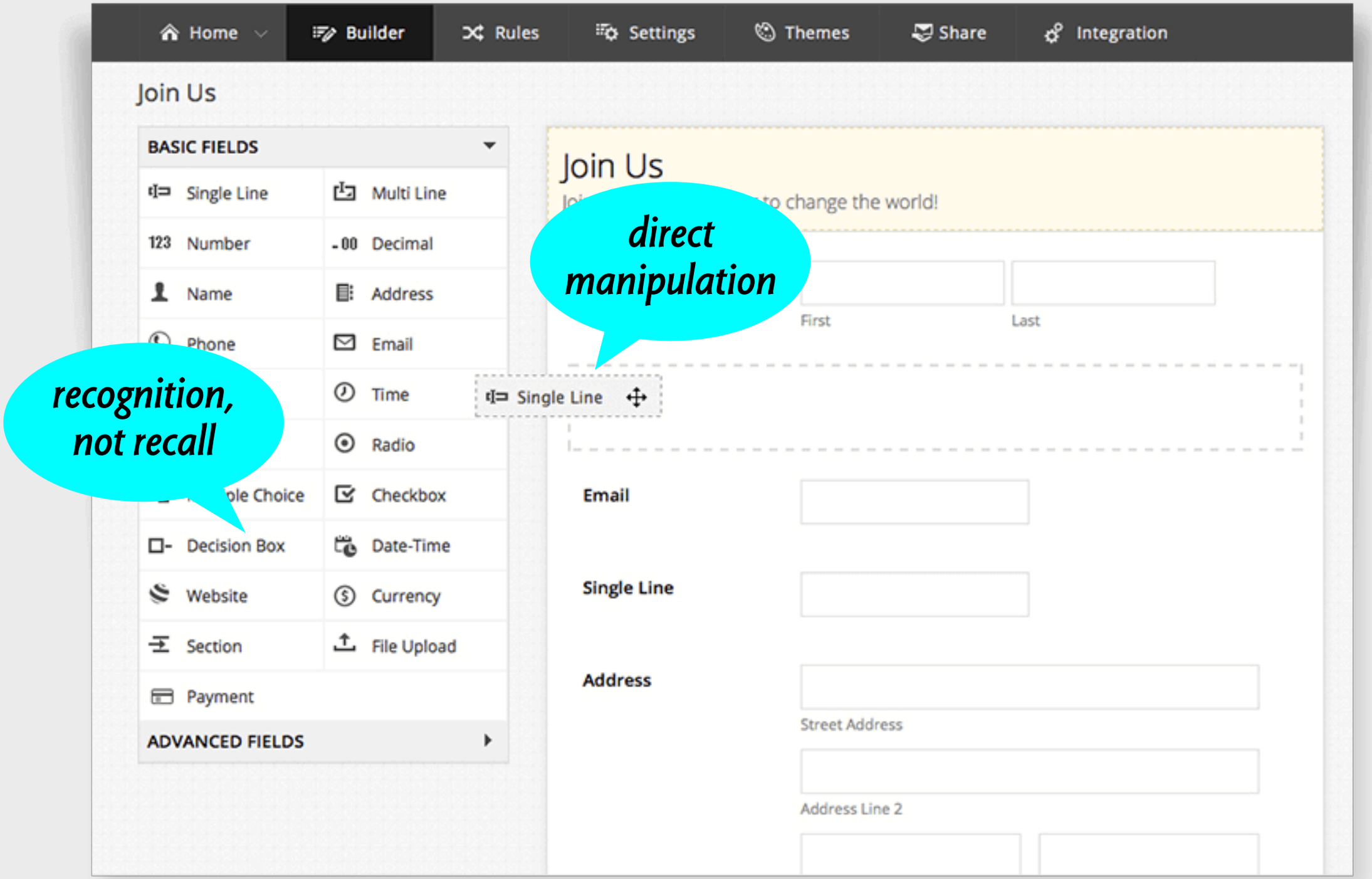
Zoho's form builder

visual editing



Zoho's form builder

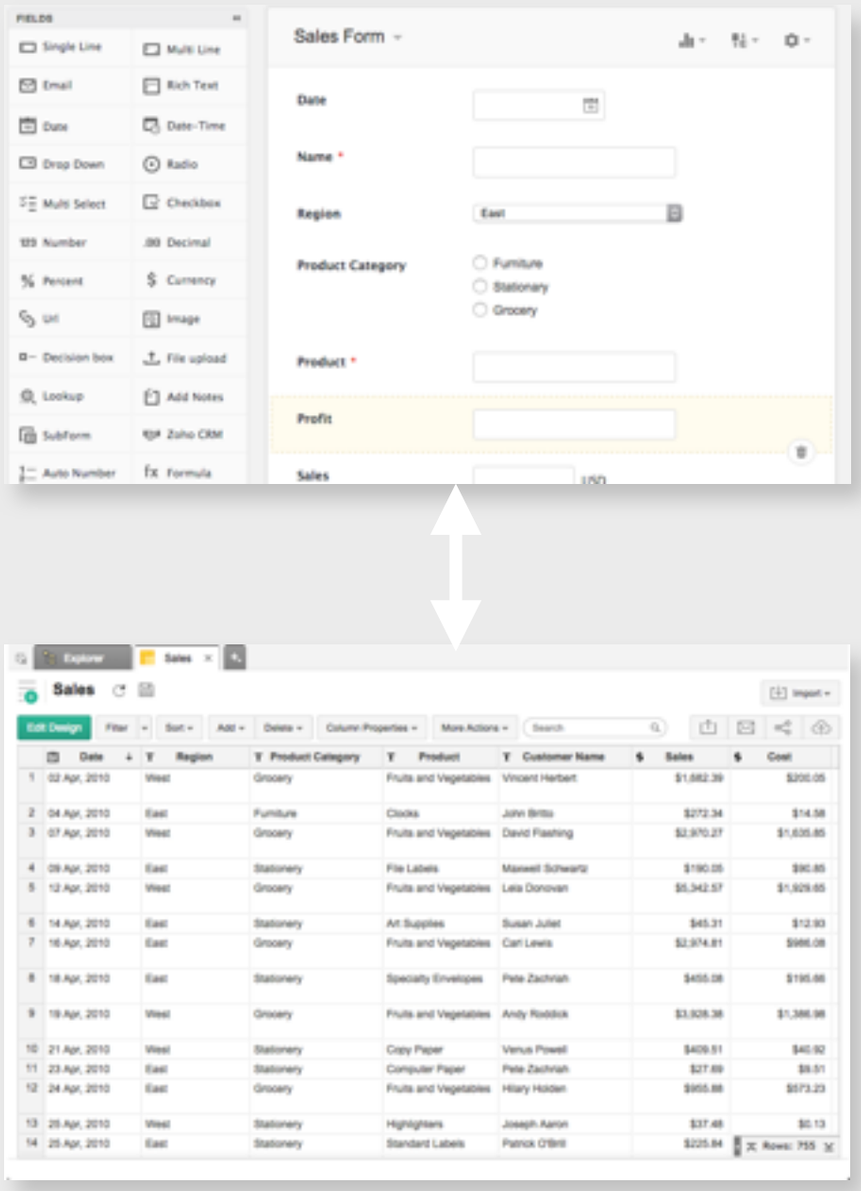
visual editing



Zoho's form builder

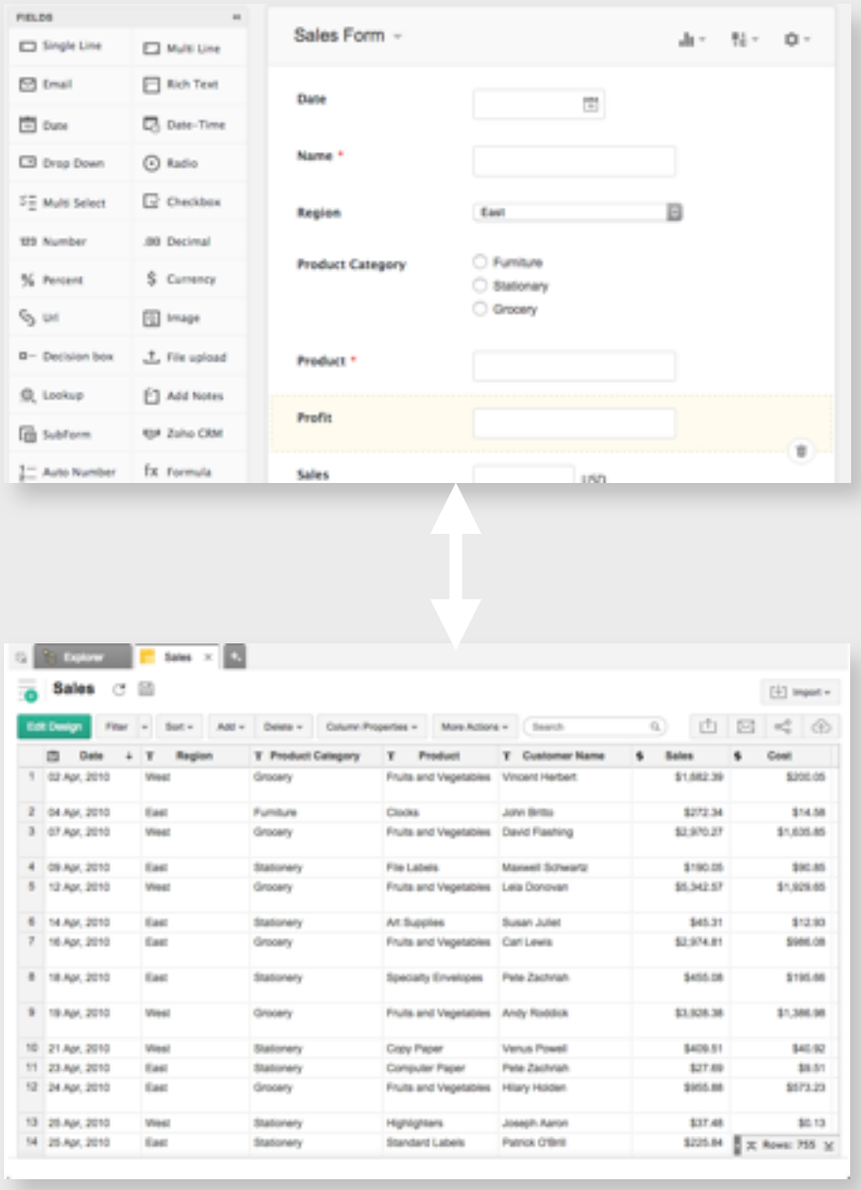
abstractions

abstractions

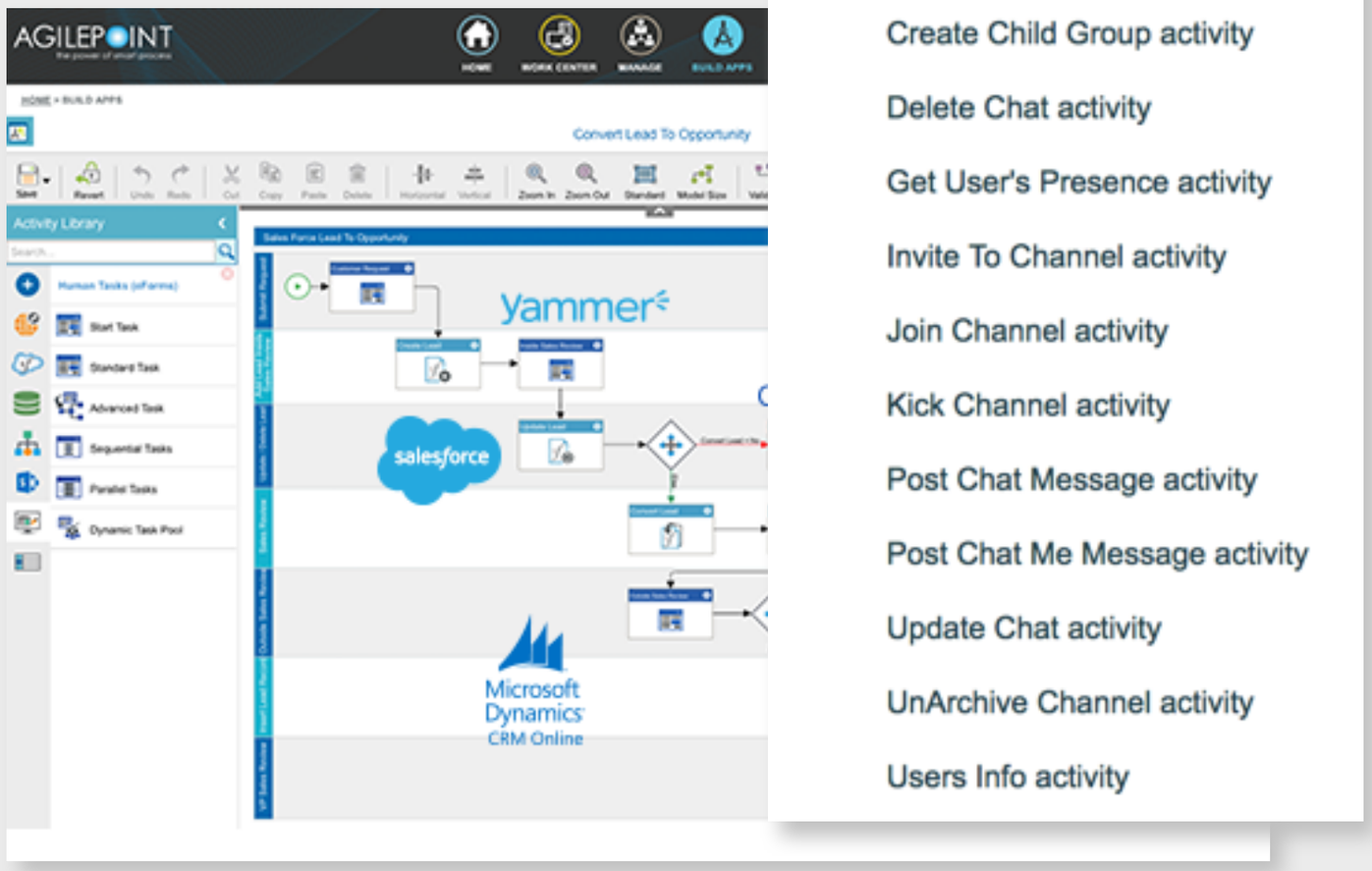


"form-tables"

abstractions

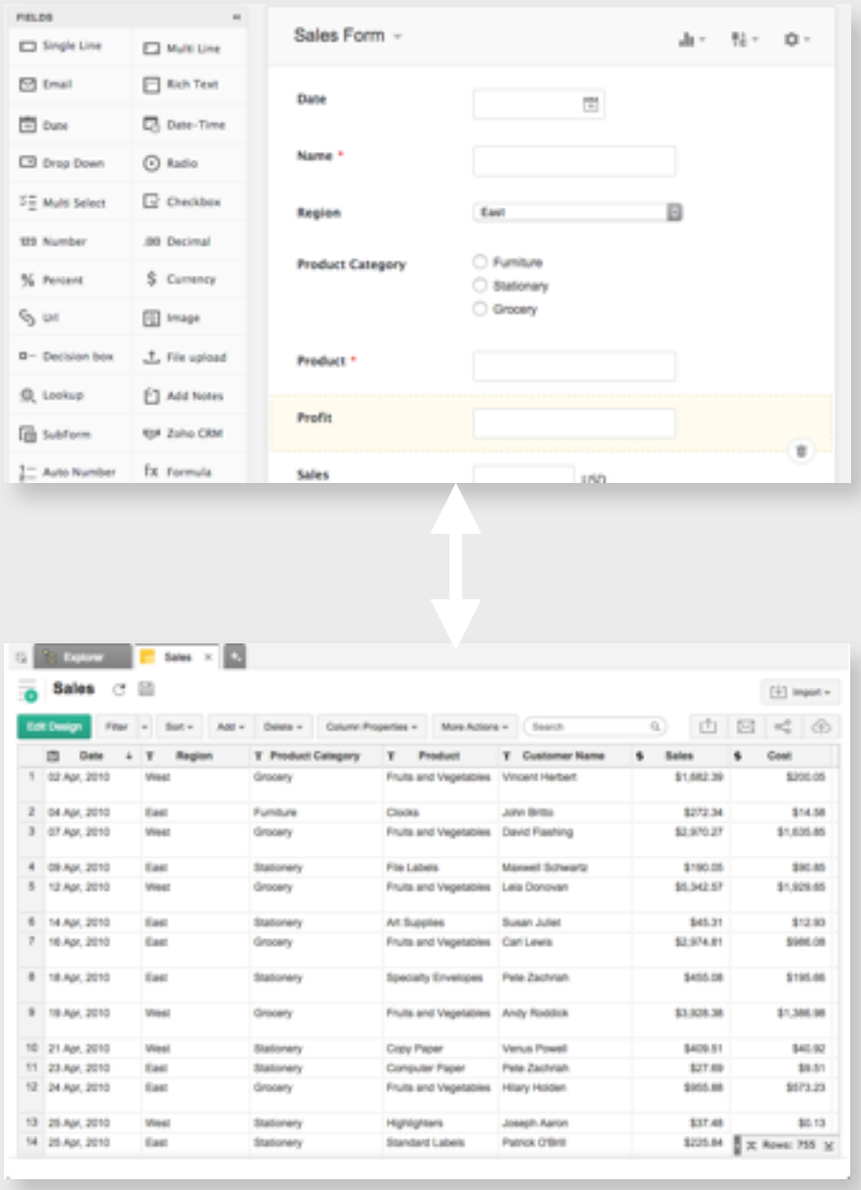


"form-tables"

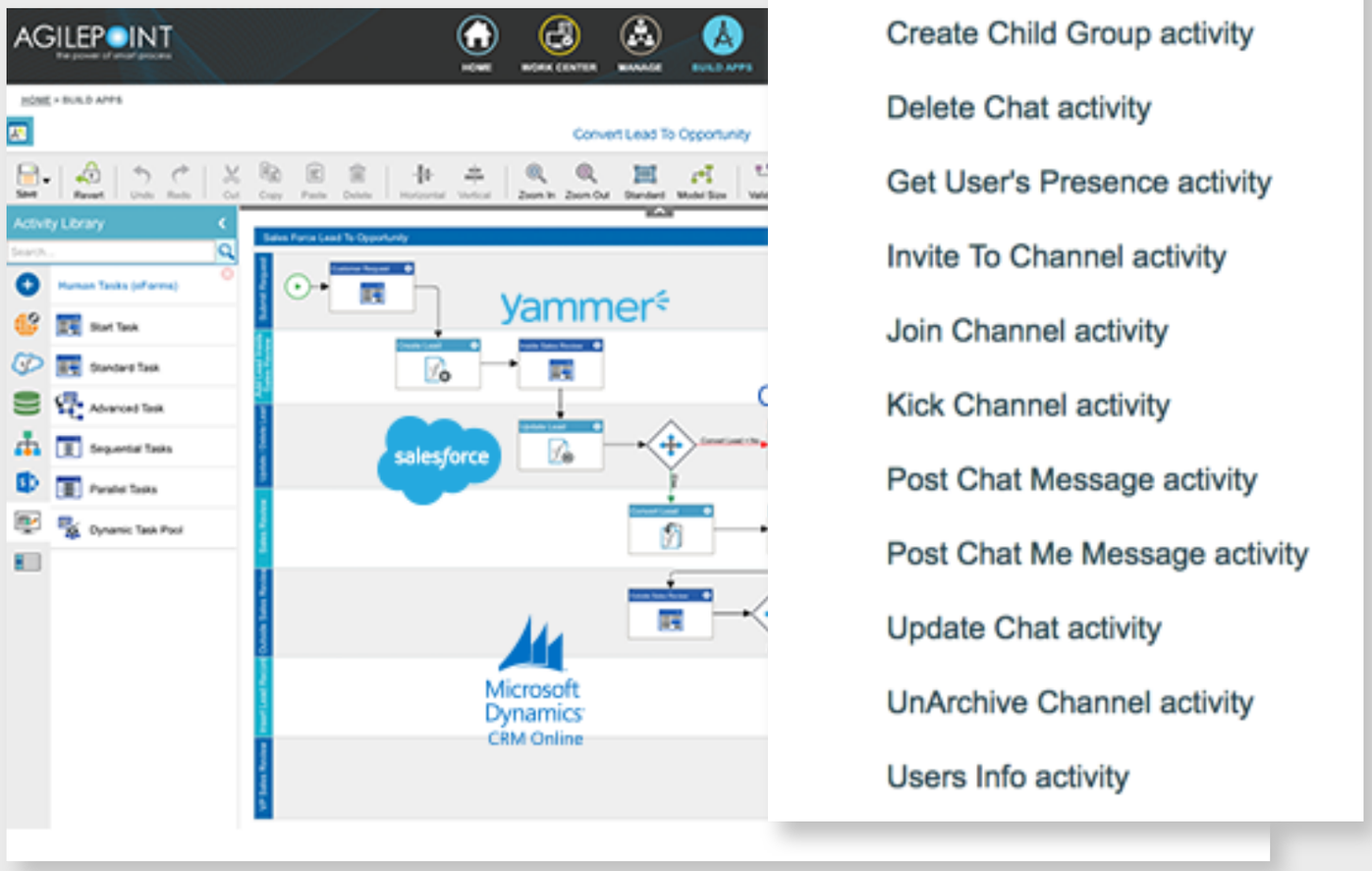


activities

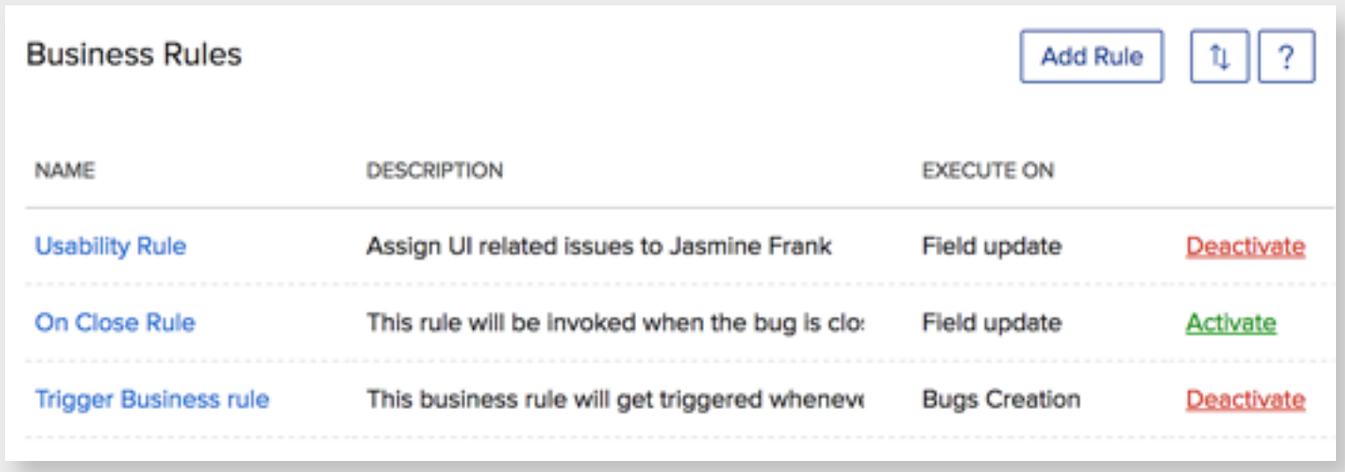
abstractions



"form-tables"

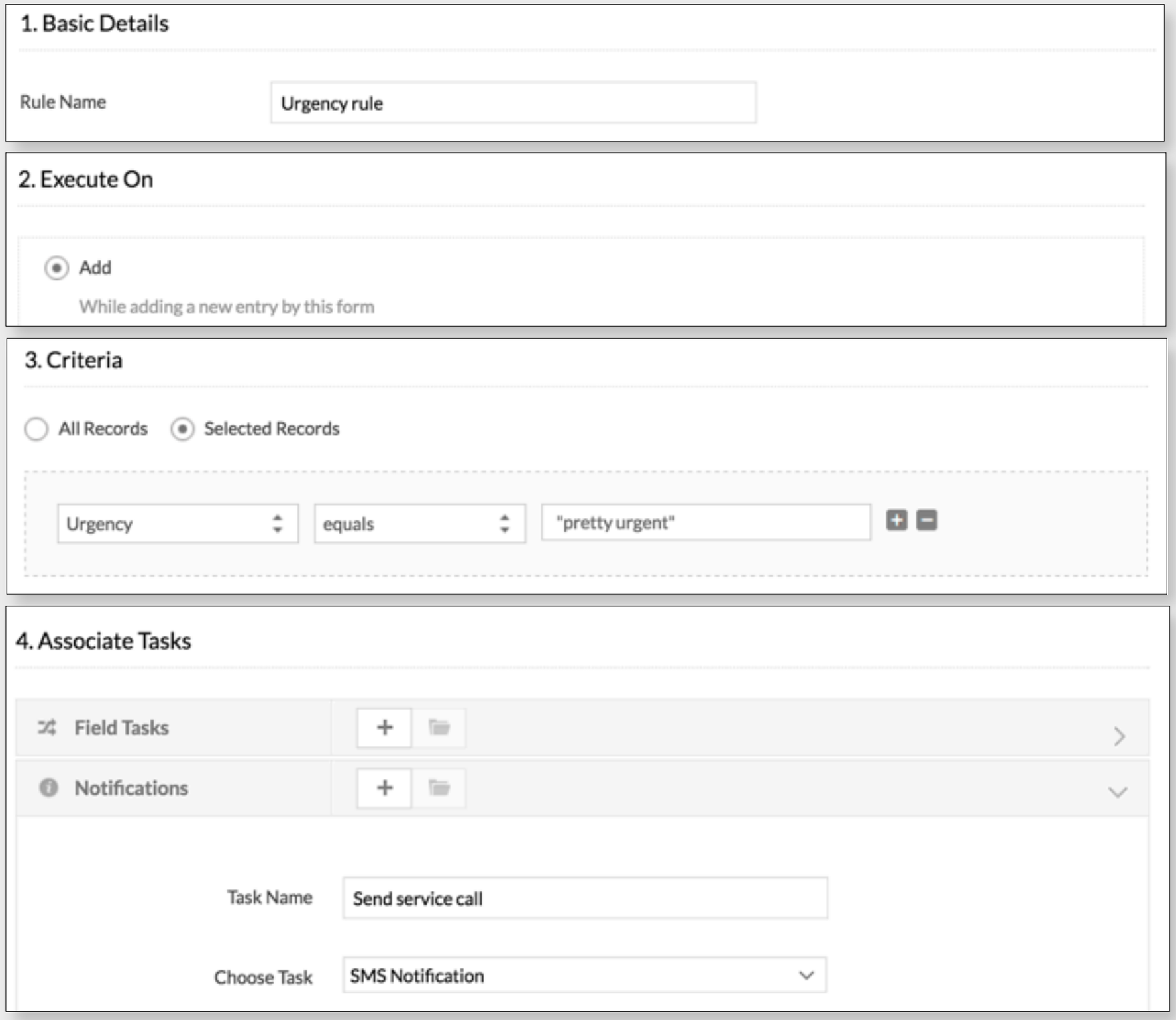


activities



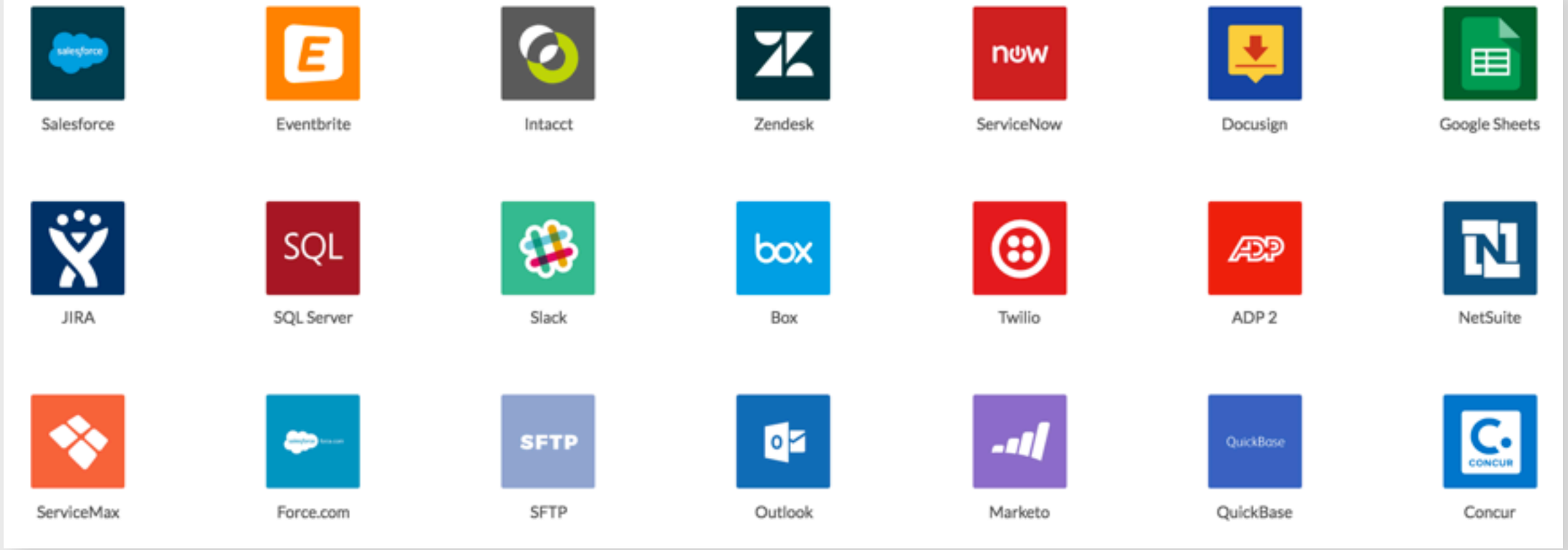
rules

declarative code



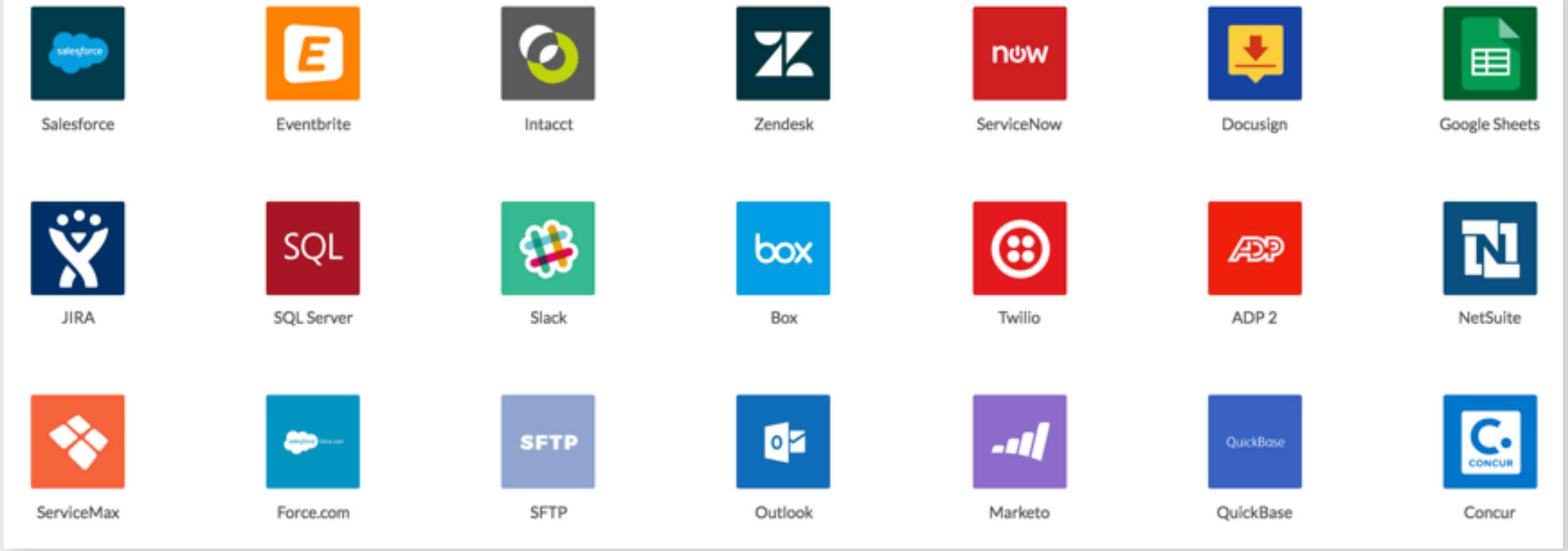
Zoho's rules & tasks

integration



Workato's integrated backends

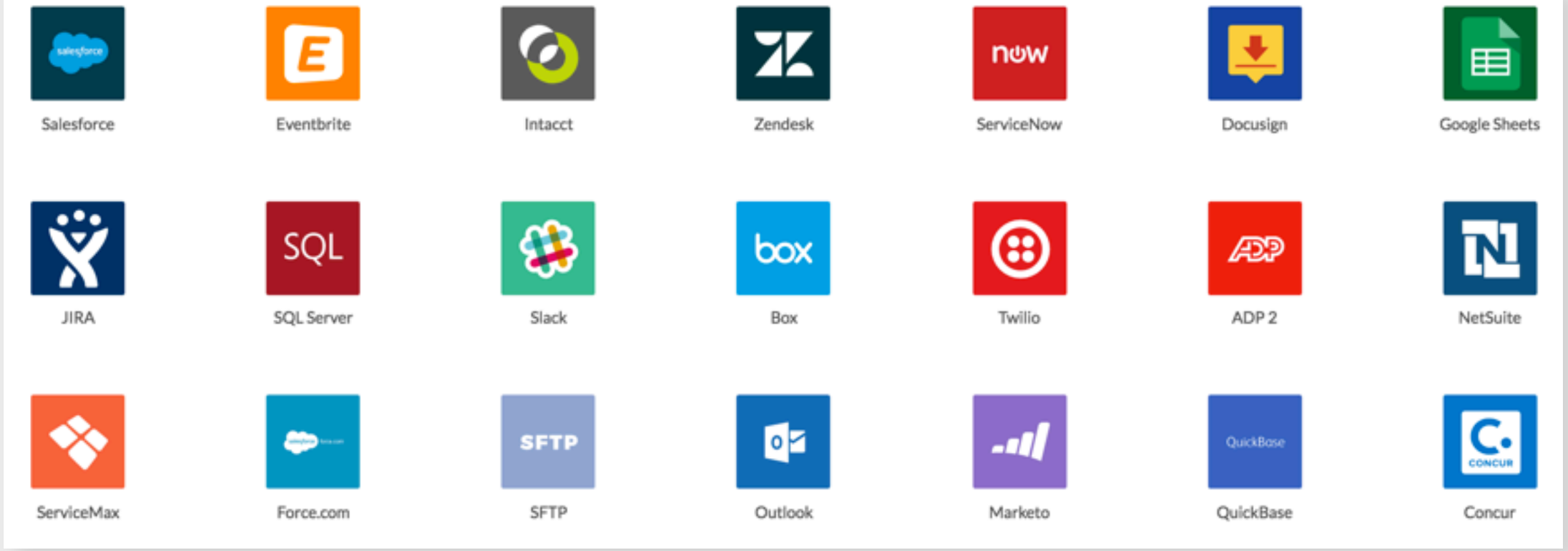
integration



Workato's integrated backends

cloud recipes: Zapier, IFTTT, Workato

integration

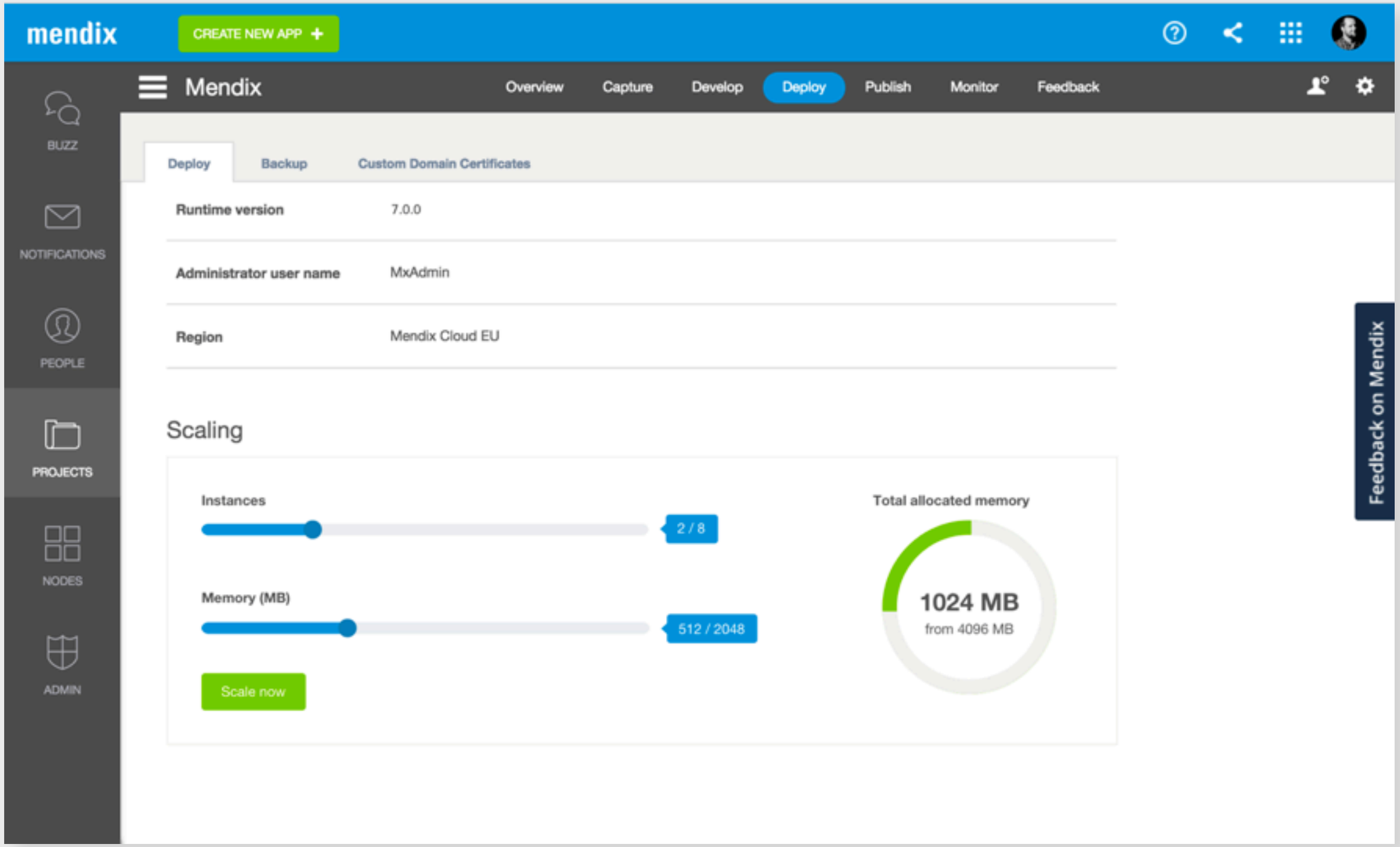


Workato's integrated backends

cloud recipes: Zapier, IFTTT, Workato

mobile apps: Appery.io, SkyGiraffe, Appian QuickApps

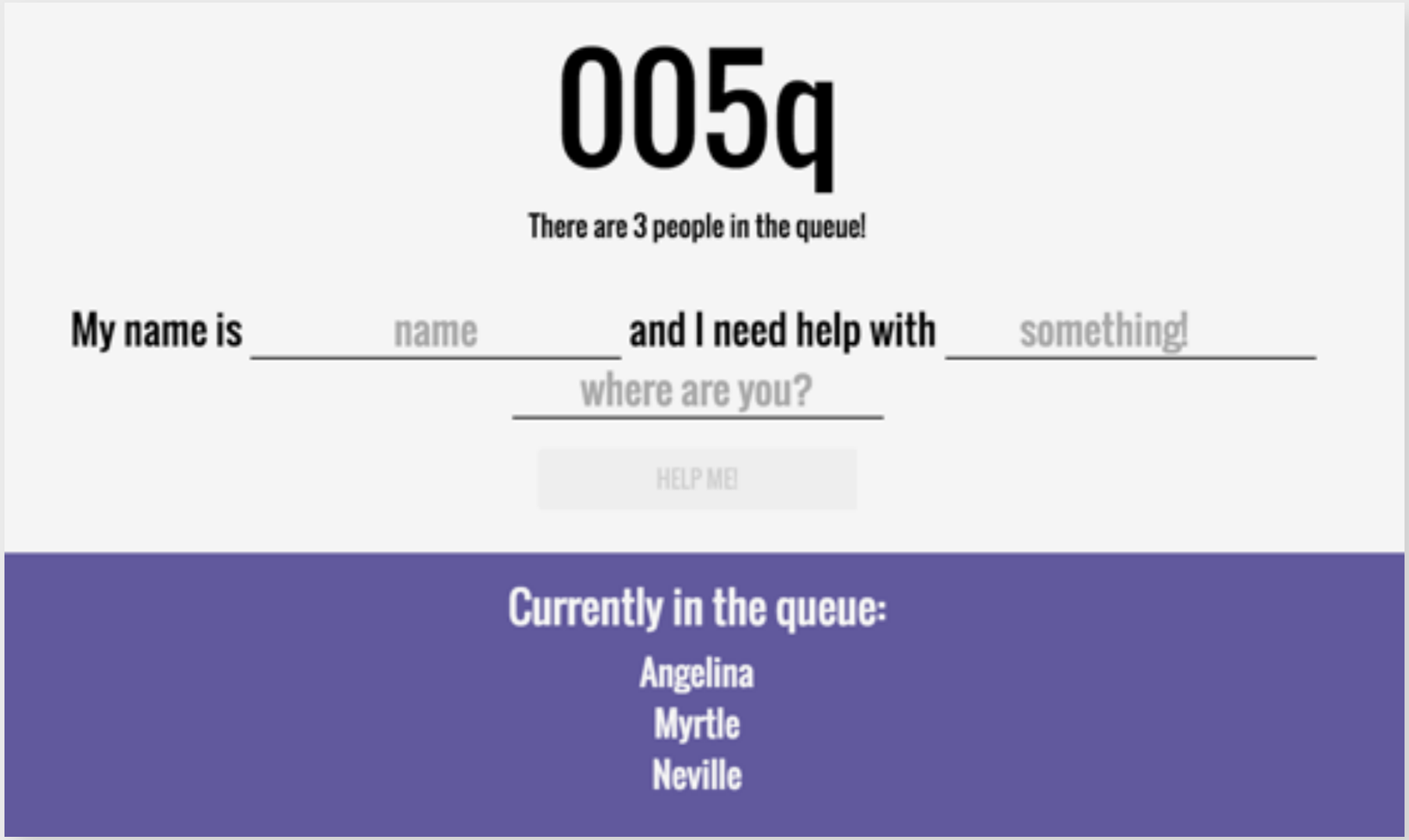
easy deployment



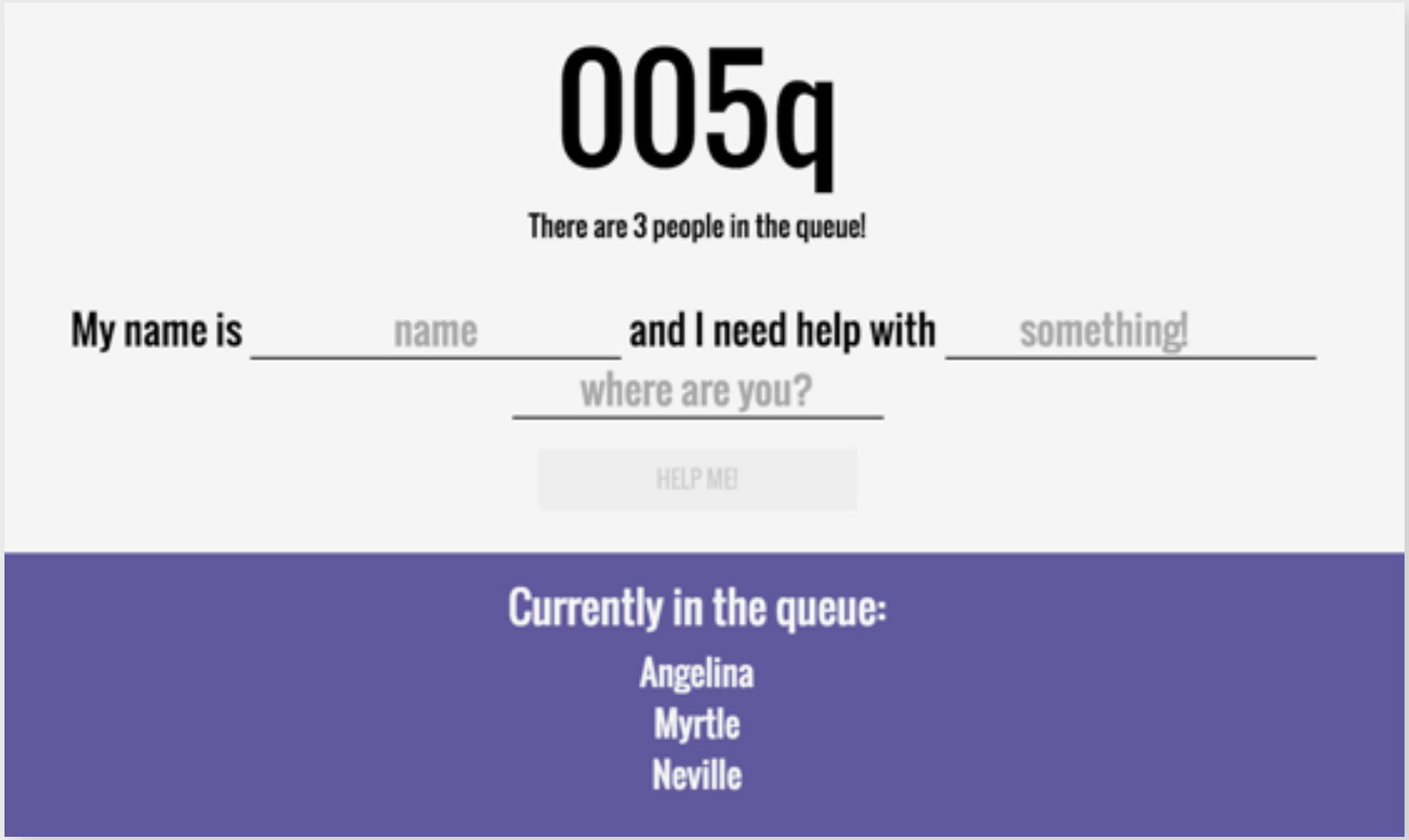
Mendix's AWS-based deployment

not yet in paradise

an example: hackathon Q

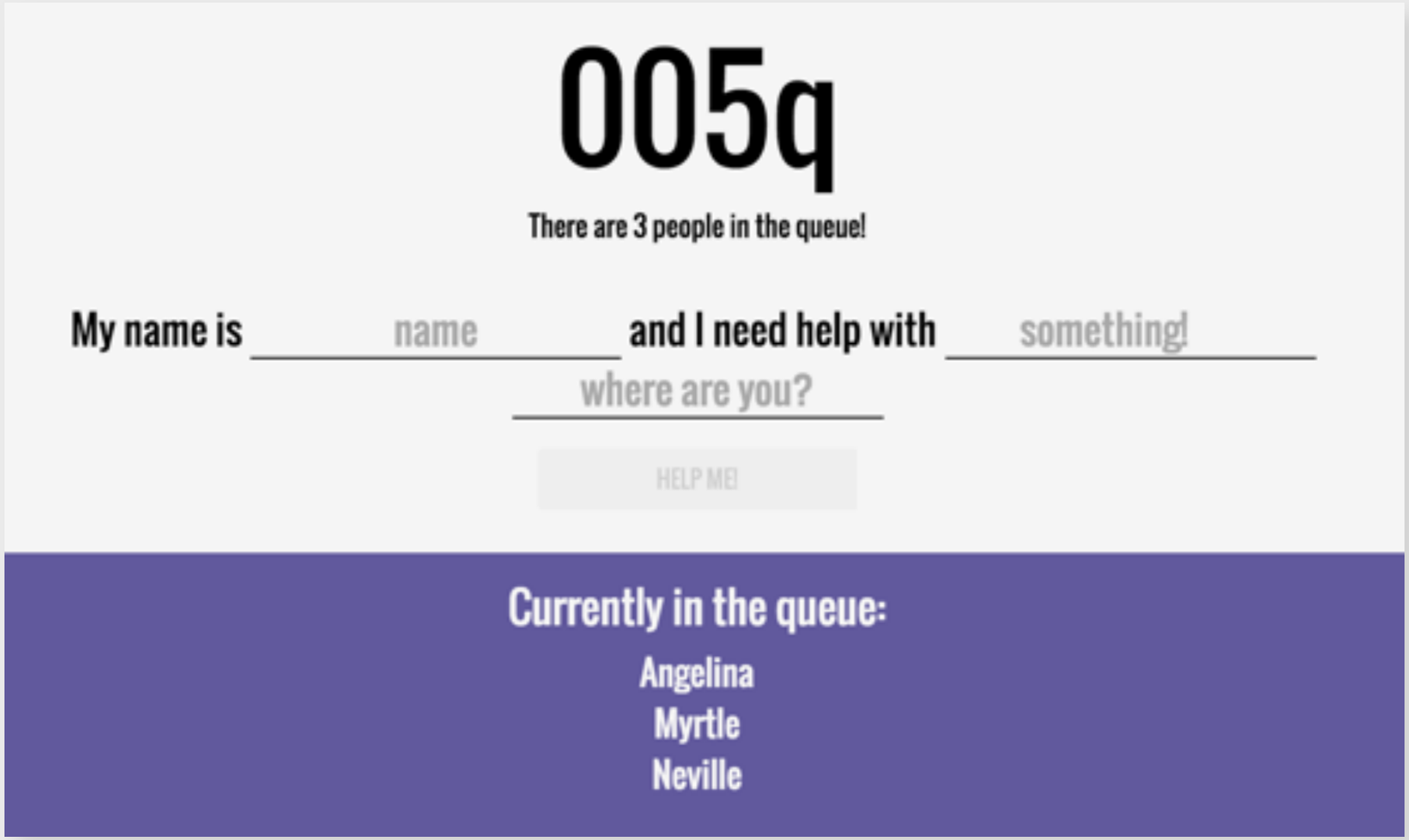


an example: hackathon Q



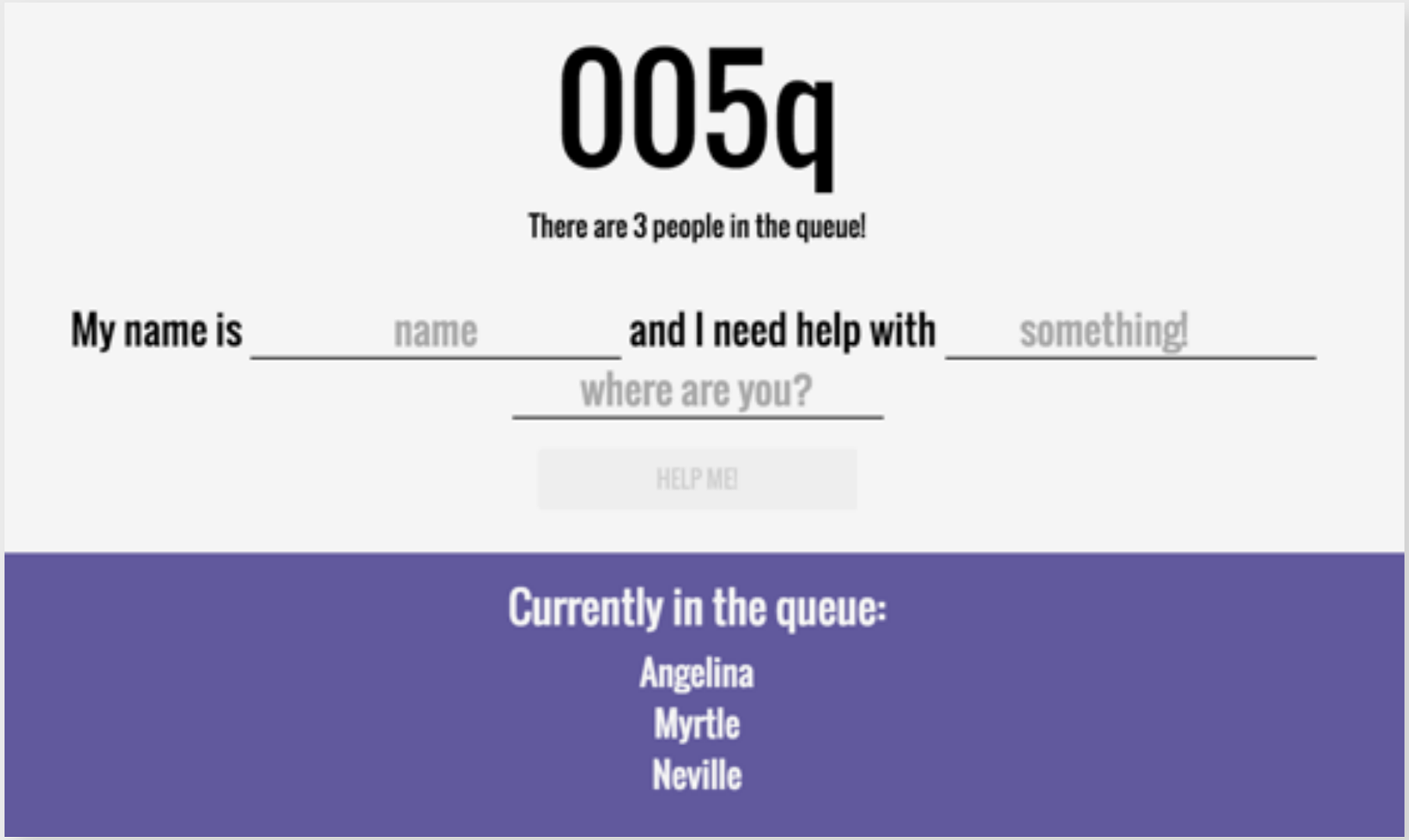
mentors register skills

an example: hackathon Q



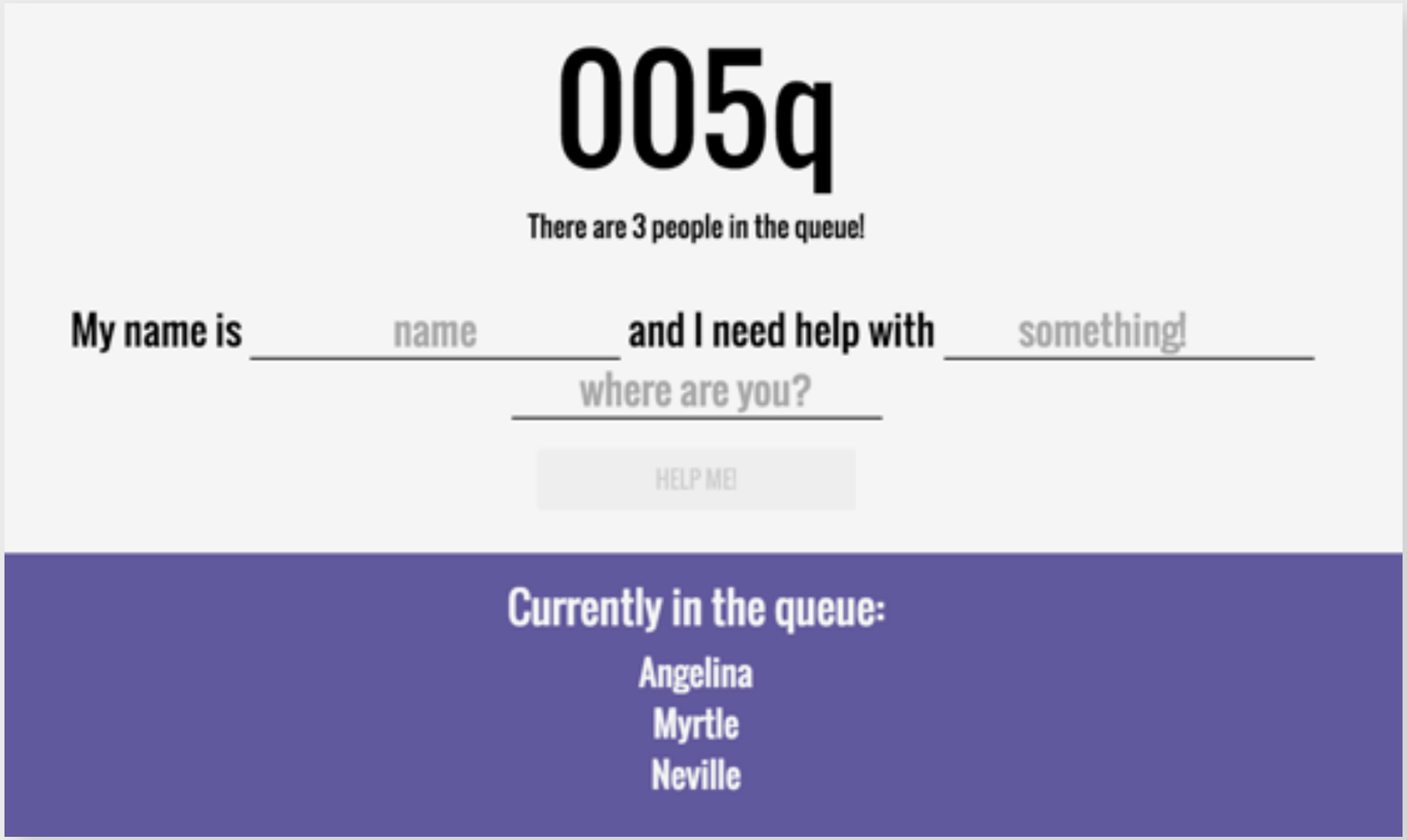
mentors register skills
participants request a skill

an example: hackathon Q



mentors register skills
participants request a skill
mentor assigns calls

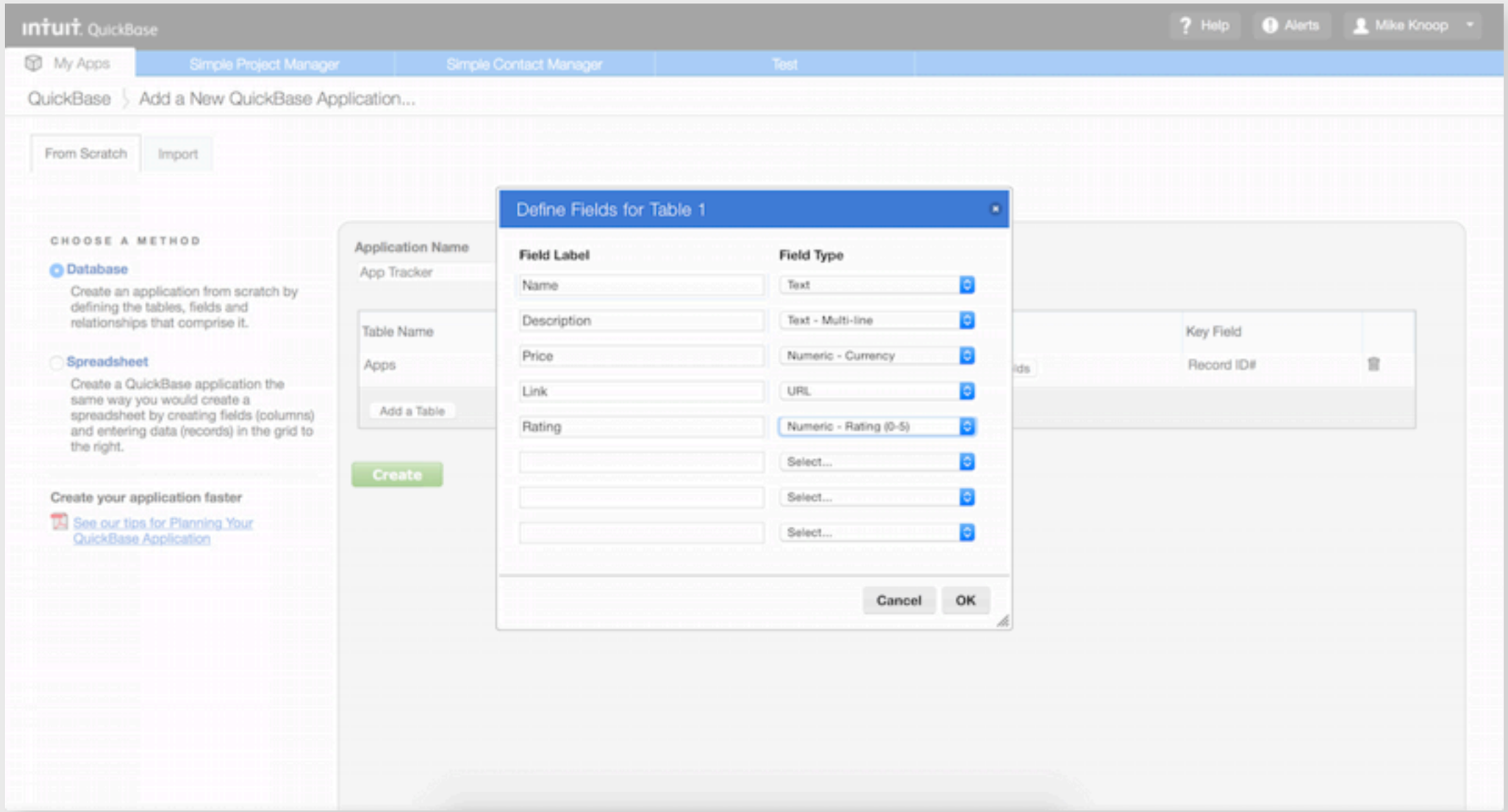
an example: hackathon Q



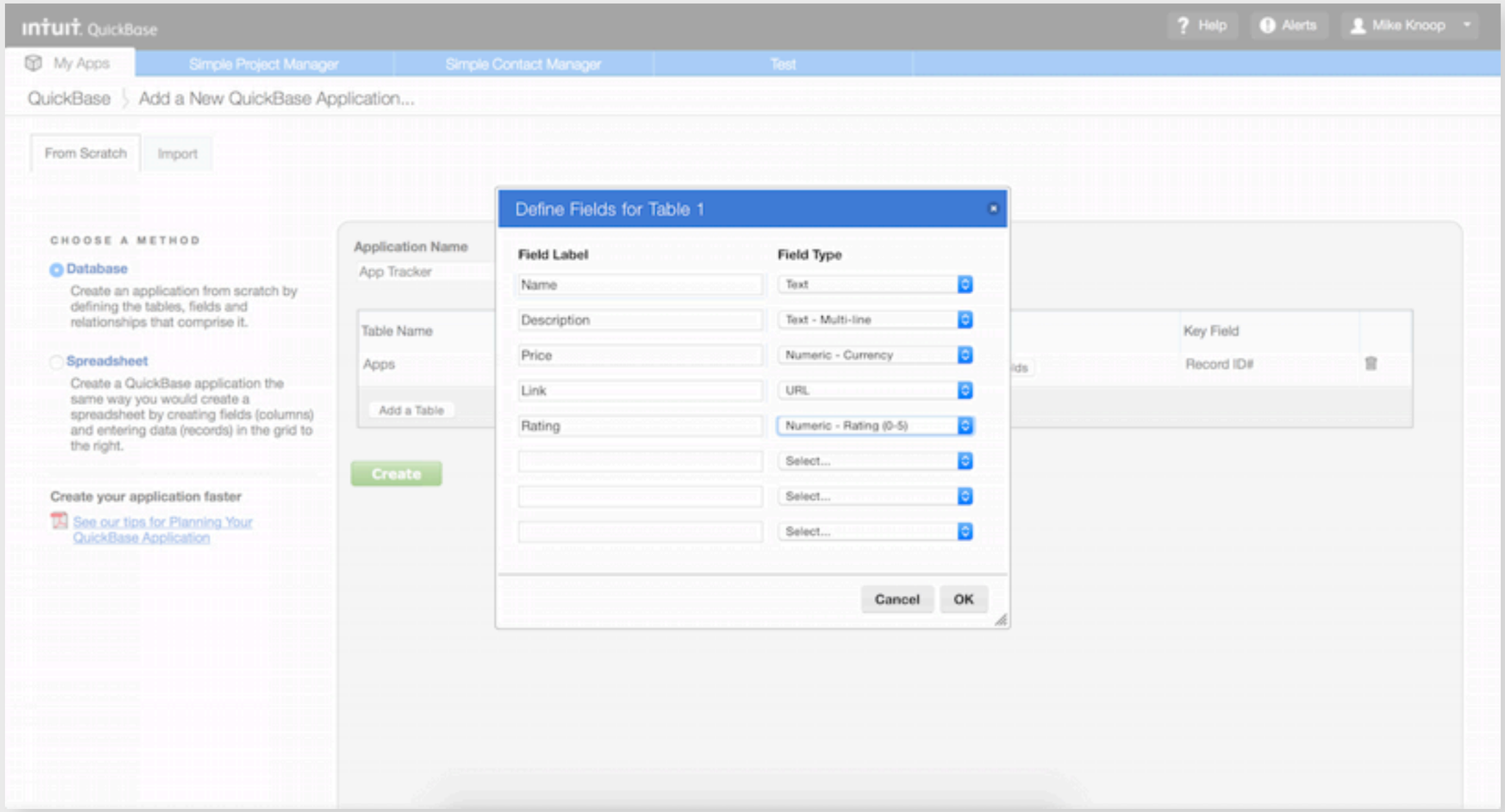
mentors register skills
participants request a skill
mentor assigns calls

tables: skills, mentors, calls
report: calls active/assigned
forms: request, offer, assign

Quickbase

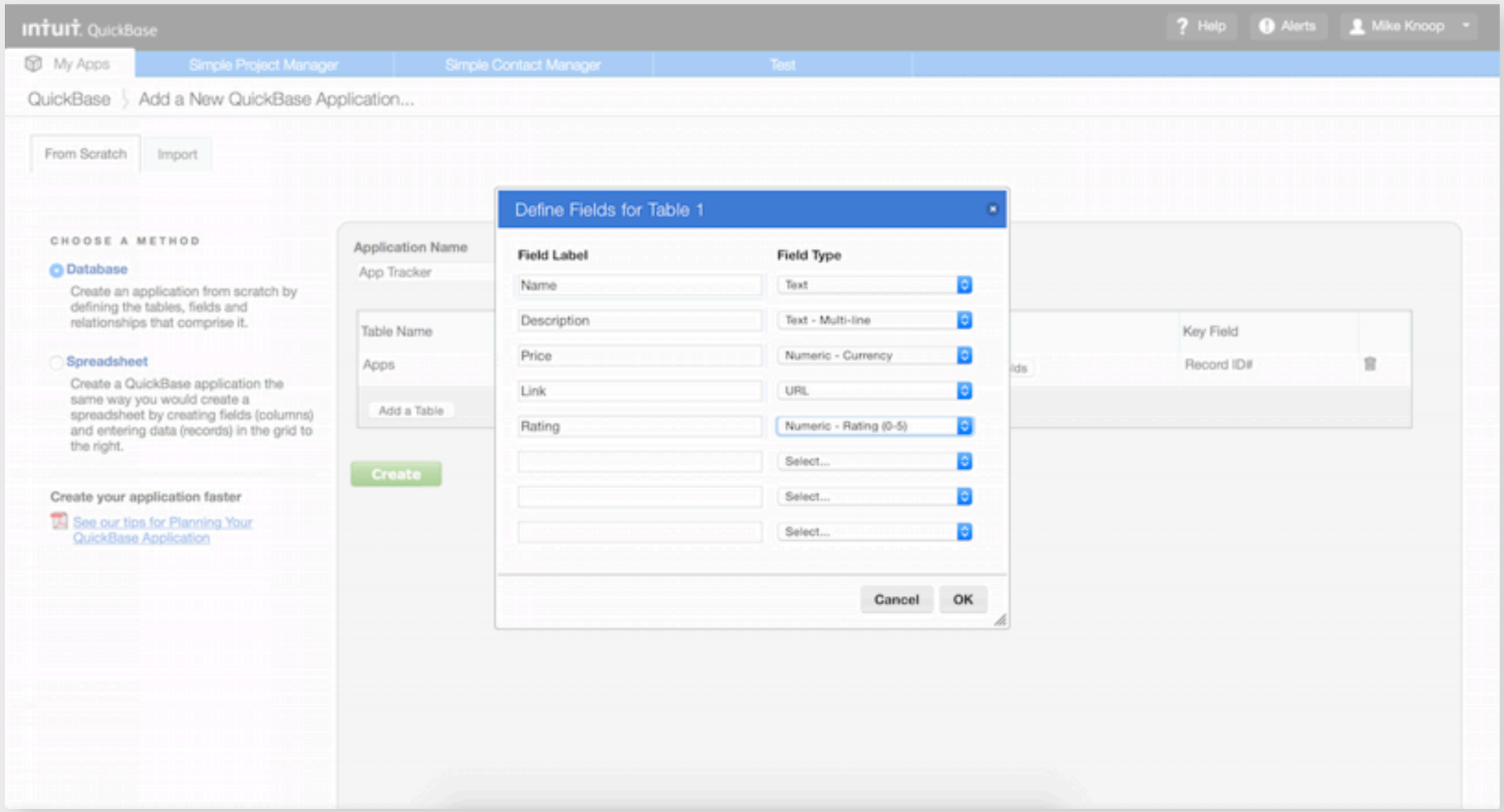

intuit QuickBase
Help
Alerts
Mike Knoop
My Apps
Simple Project Manager
Simple Contact Manager
Test
QuickBase
Add a New QuickBase Application...
From Scratch
Import
CHOOSE A METHOD
Database
Create an application from scratch by defining the tables, fields and relationships that comprise it.
Spreadsheet
Create a QuickBase application the same way you would create a spreadsheet by creating fields (columns) and entering data (records) in the grid to the right.
Create your application faster
See our tips for Planning Your QuickBase Application
Application Name
App Tracker
Table Name
Apps
Add a Table
Create
Key Field
Record ID#
Define Fields for Table 1
Field Label
Field Type
Name
Text
Description
Text - Multi-line
Price
Numeric - Currency
Link
URL
Rating
Numeric - Rating (0-5)
Select...
Select...
Cancel
OK

Quickbase



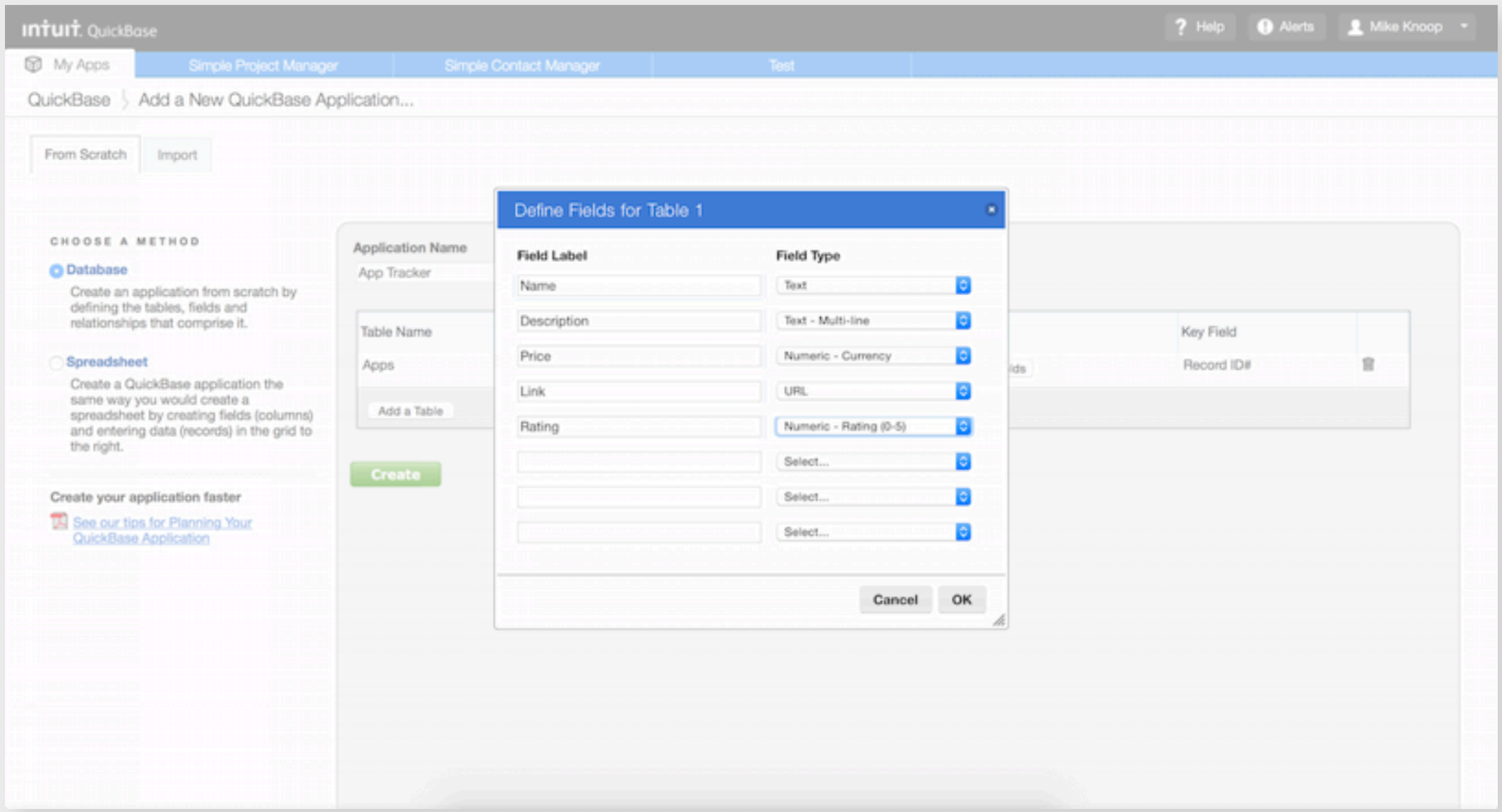
tricky: had to provide reverse mapping from skills to mentors

Quickbase



tricky: had to provide reverse mapping from skills to mentors
not possible: allow requests for more than one skill

Quickbase



tricky: had to provide reverse mapping from skills to mentors
not possible: allow requests for more than one skill
needed custom code: assign call to current user

what's going on?

what's going on?

form-table approach
ad hoc query language
no underlying calculus

what's going on?

form-table approach
ad hoc query language
no underlying calculus

some basic app features
may require custom code
or not be expressible at all

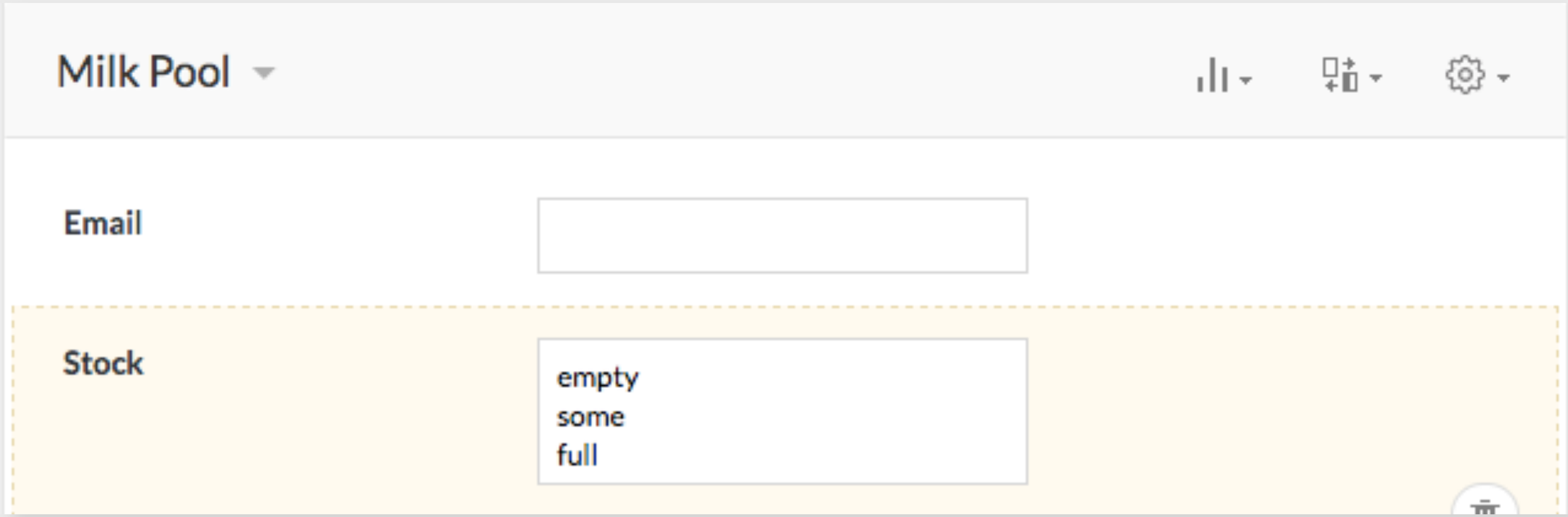
what's going on?

form-table approach
ad hoc query language
no underlying calculus

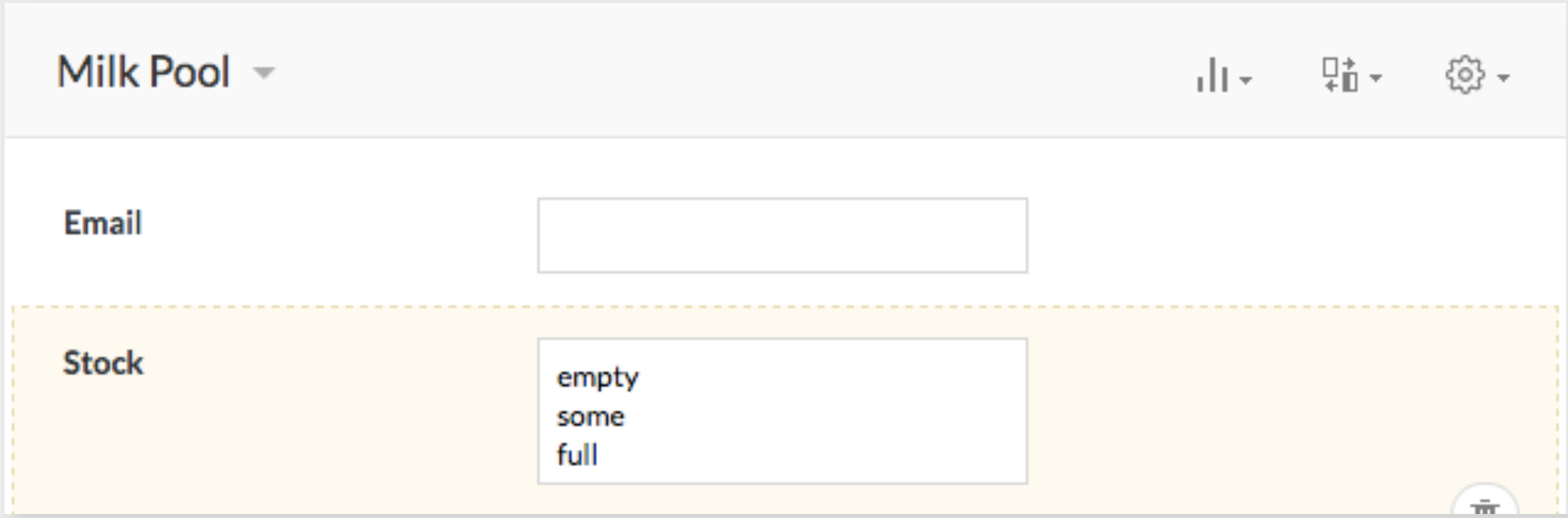
some basic app features
may require custom code
or not be expressible at all

biggest problem?
hard to predict until you try

encountering limitations



encountering limitations



let's make a rule

if user reports out of milk
then send message

not uniformly visual

not uniformly visual

not uniformly visual

1. Basic Details

Rule Name: Milk low

3. Criteria

All Records | Selected Records

Stock | equals | ""

not uniformly visual

1. Basic Details

Rule Name: Milk low

3. Criteria

All Records Selected Records

must recall field values

Stock | equals | ""

not uniformly visual

1. Basic Details

Rule Name: Milk low

3. Criteria

All Records / Selected Records

Stock | equals | "low"

not uniformly visual

1. Basic Details

Rule Name: Milk low

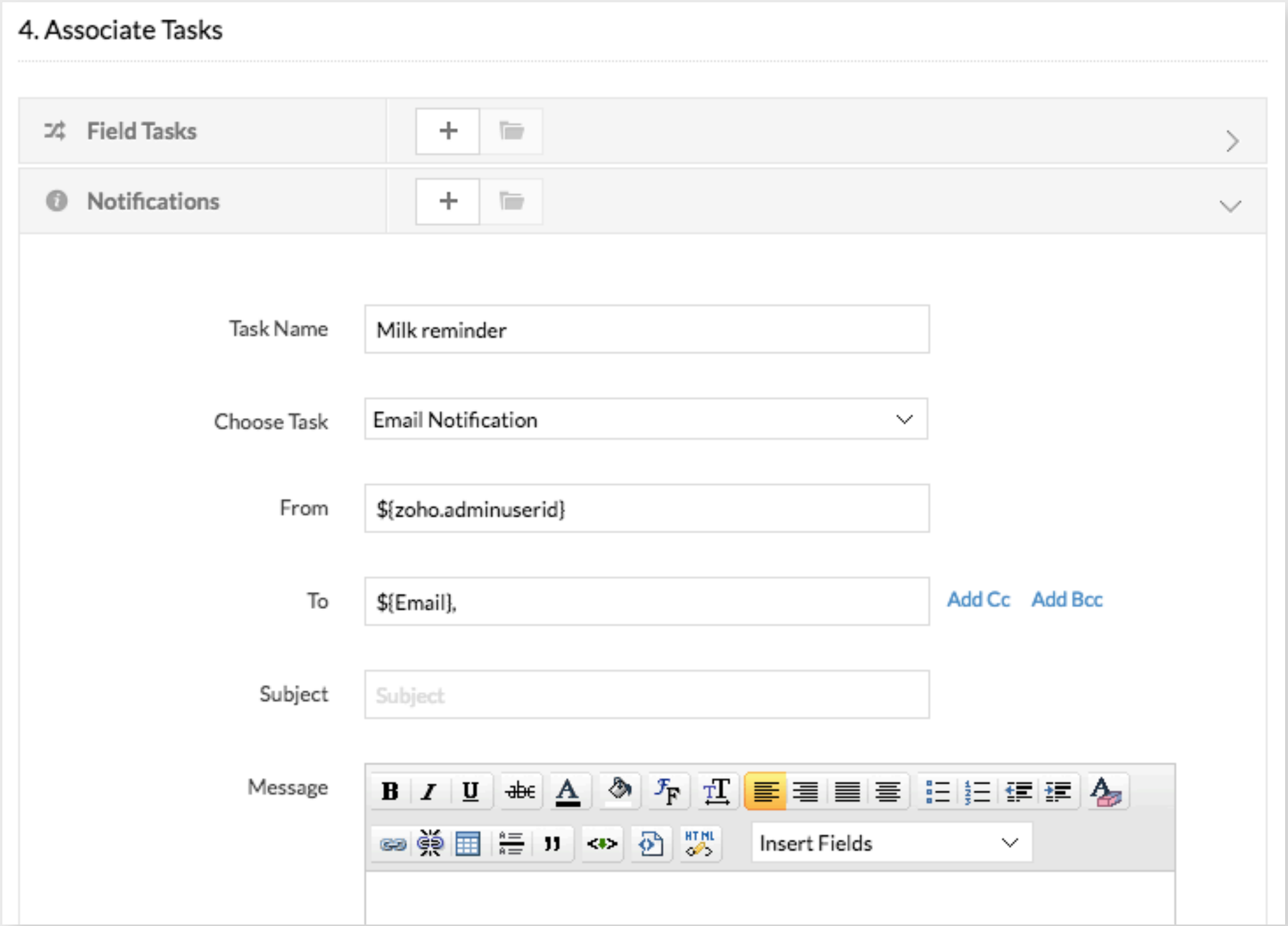
3. Criteria

All Records | Selected Records

bad value, no warning

Stock | equals | "low"

not uniformly expressive



not uniformly expressive

4. Associate Tasks

Field Tasks

Notifications

Task Name: Milk reminder

Choose Task: Email Notification

From: ${zoho.adminuserid}

To: ${Email},

Add Cc Add Bcc

Subject: Subject

Message

Insert Fields

can use variable here

not uniformly expressive

4. Associate Tasks

Field Tasks

Notifications

Task Name: Milk reminder

Choose Task: Email Notification

From: ${zoho.adminuserid}

To: ${Email},

Subject: Subject

Message

Insert Fields

can use variable here

but not here

welcome back, VBA?

All Functions

The following table is a searchable listing of all Appian functions.

- Functions in this table are sorted by *category*, *sub-category*, then function *name*.
- By default, this table shows the function name, and an example where available. You can use the *col* specific columns.
- You have *filtering options* along the top-right side of the table, where you can filter the table by funct filter on function name and category.
- For more detailed information about a particular function, click the function name to go to its page.

DESCRIPTION | SYNTAX | EXAMPLE | RESULT | COLLAPSE ALL

Name	Example
Array Used within your expressions to manipulate, insert, and/or select values from arrays.	
append()	`append({10, 20, 30}, 99)`
index()	`index({10, 20, 30}, 2, 1)`
insert()	`insert({10, 20, 30, 40}, 99, 1)`
joinarray()	`joinarray({1, 2, 3, 4}, "|")`
ldrop()	`ldrop({10, 20, 30}, 1)`

Appian function API

some new research

two research projects

two research projects

aim

new approach to low-code

two research projects

aim
new approach to low-code

target
“community applications”
too complex for Drupal
too simple for full stack

two research projects

aim
new approach to low-code

target
“community applications”
too complex for Drupal
too simple for full stack

approach
language first, wizards later
declarative & expressive

a new layout language

a new layout language

written by non-programmers
flexible visual design
cross-platform & responsive
rich hierarchical data model

a new layout language

written by non-programmers
flexible visual design
cross-platform & responsive
rich hierarchical data model

HTML only for data instances
can't describe schemas
need JavaScript to read/write

mavo

by Lea Verou & David Karger

written by non-programmers
flexible visual design
cross-platform & responsive
rich hierarchical data model

mavo

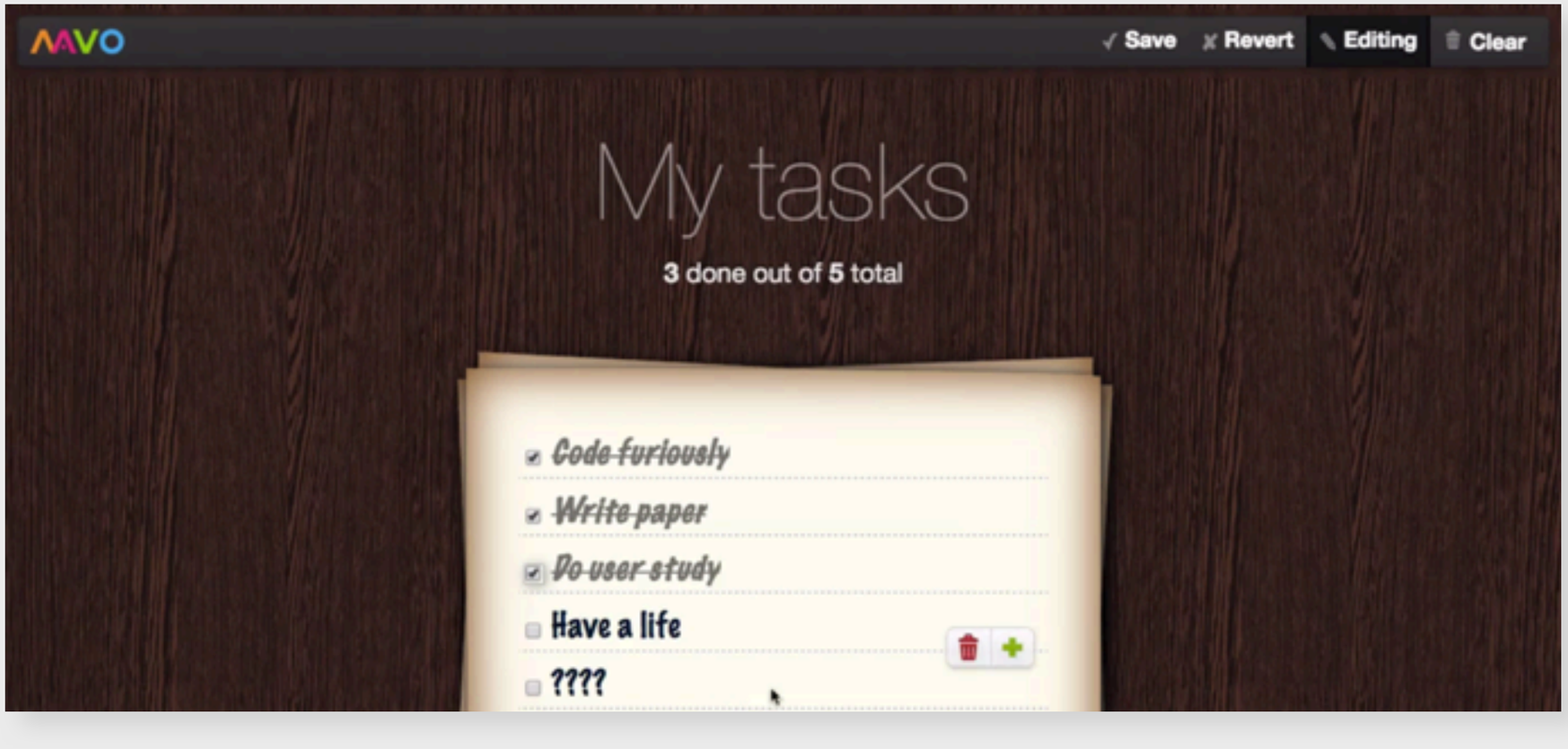
by Lea Verou & David Karger

written by non-programmers
flexible visual design
cross-platform & responsive
rich hierarchical data model

use HTML instance as schema
make elements editable
read/write to server for free

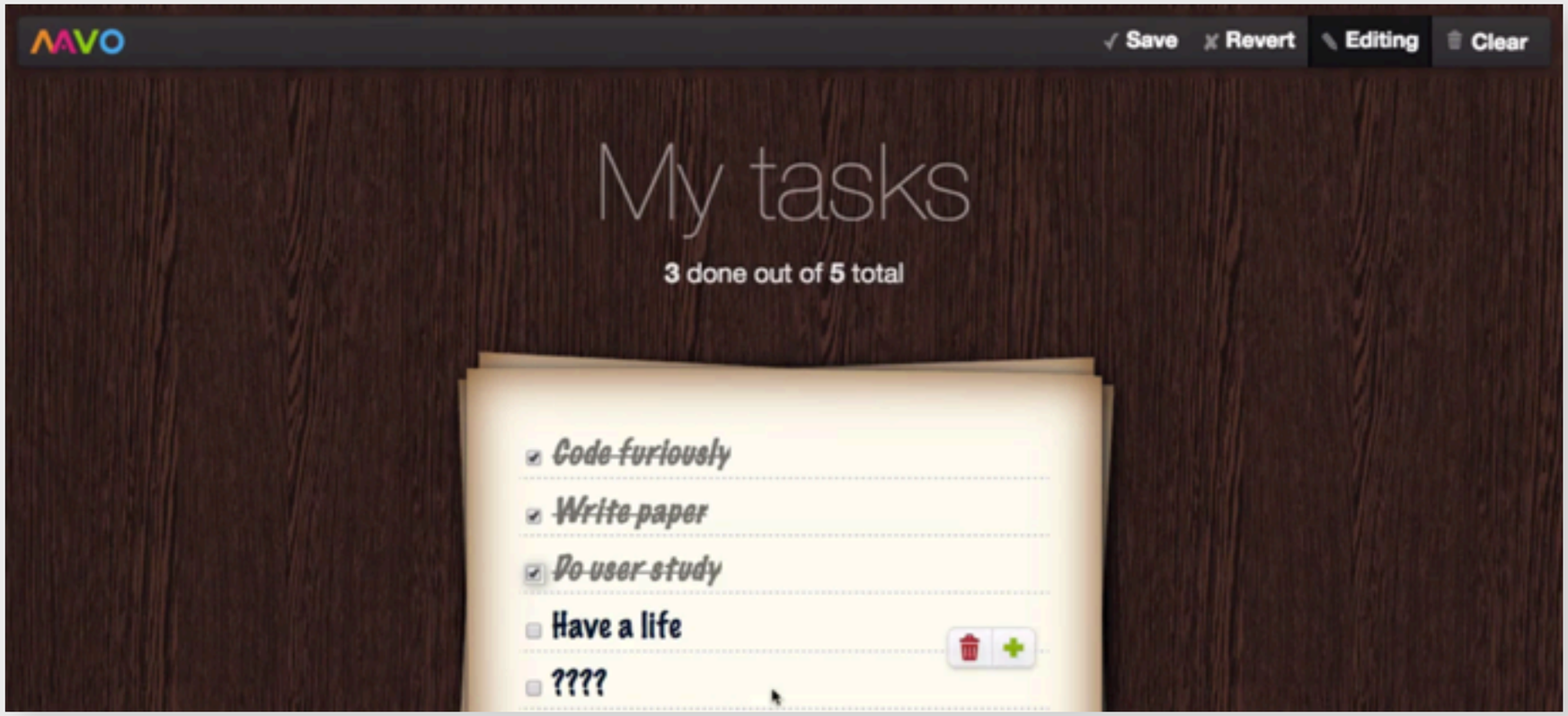
to-do in Mavo

the app



to-do in Mavo

the app

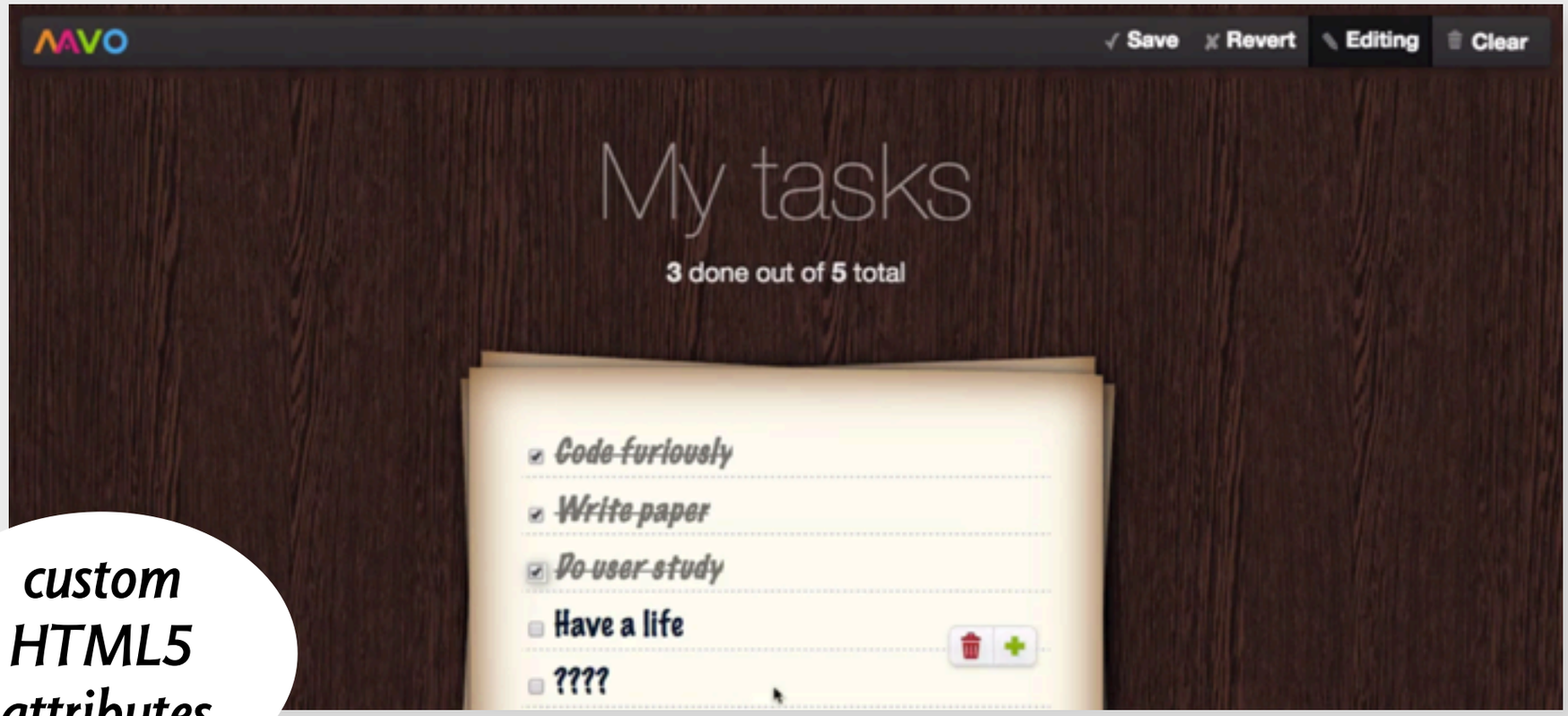


the code

```
<body data-store="local">
    <h1>My tasks</h1>
    <p>[count(done)] done of [count(task)] total</p>

    <ul>
        <li property="task" data-multiple>
            <input property="done" type="checkbox" />
            <span property="taskTitle">Do stuff</span>
        </li>
    </ul>
</body>
```

to-do in Mavo

the app



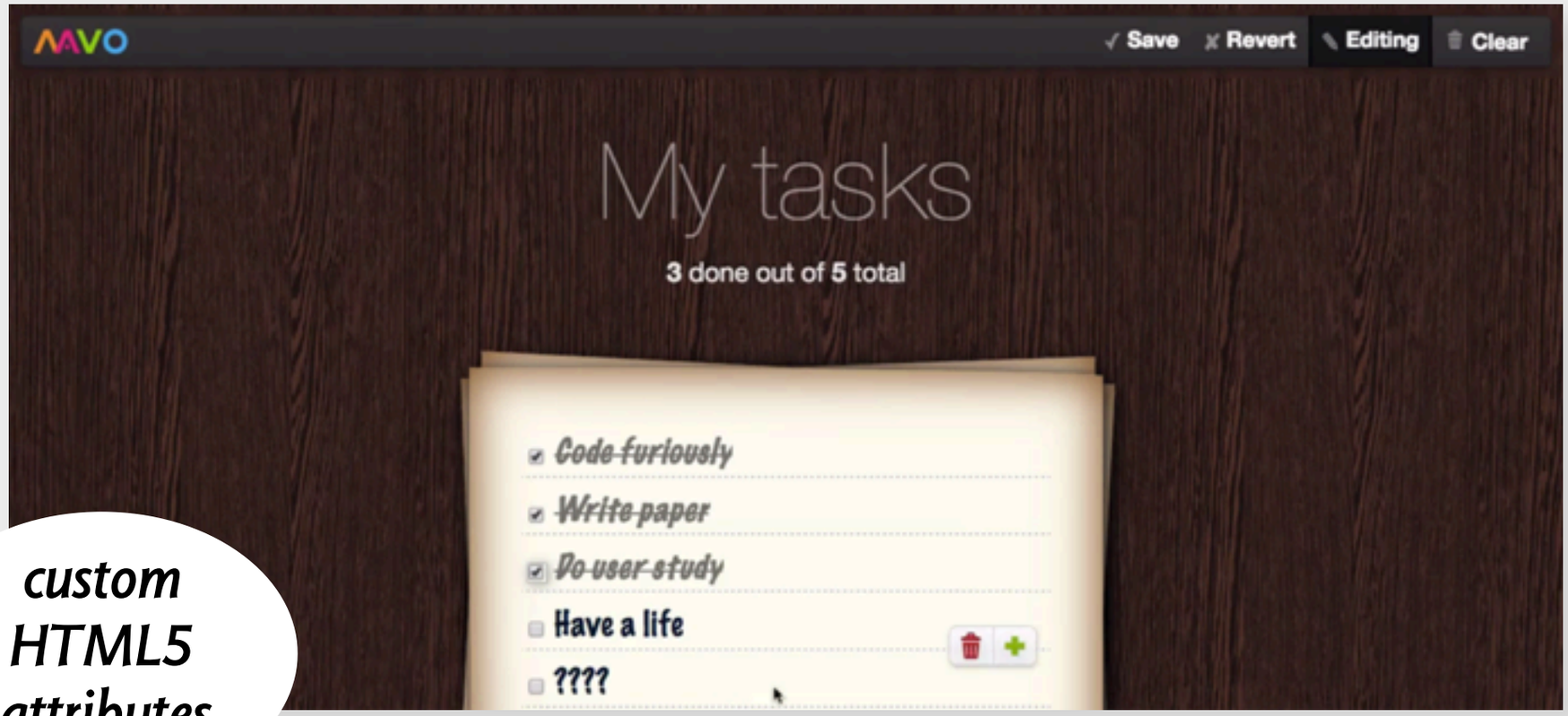
custom HTML5 attributes

the code

```
<body data-store="local">
    <h1>My tasks</h1>
    <p>[count(done)] done of [count(task)] total</p>

    <ul>
        <li property="task" data-multiple>
            <input property="done" type="checkbox" />
            <span property="taskTitle">Do stuff</span>
        </li>
    </ul>
</body>
```

to-do in Mavo

the app



custom HTML5 attributes

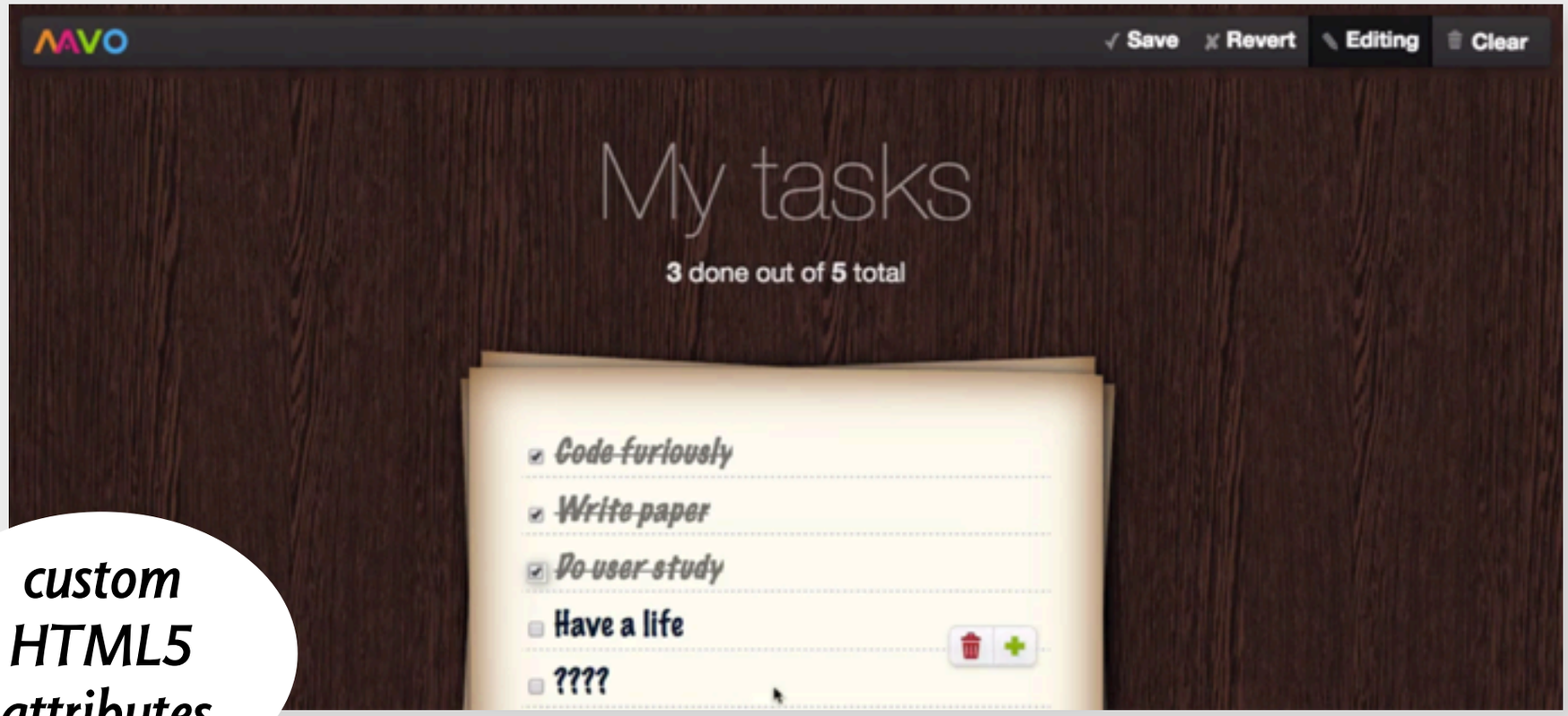
the code

```
<body data-store="local">
    <h1>My tasks</h1>
    <p>[count(done)] done of [count(task)] total</p>

    <ul>
        <li property="task" data-multiple>
            <input property="done" type="checkbox" />
            <span property="taskTitle">Do stuff</span>
        </li>
    </ul>
</body>
```

data becomes schema

to-do in Mavo

the app



custom HTML5 attributes

the code

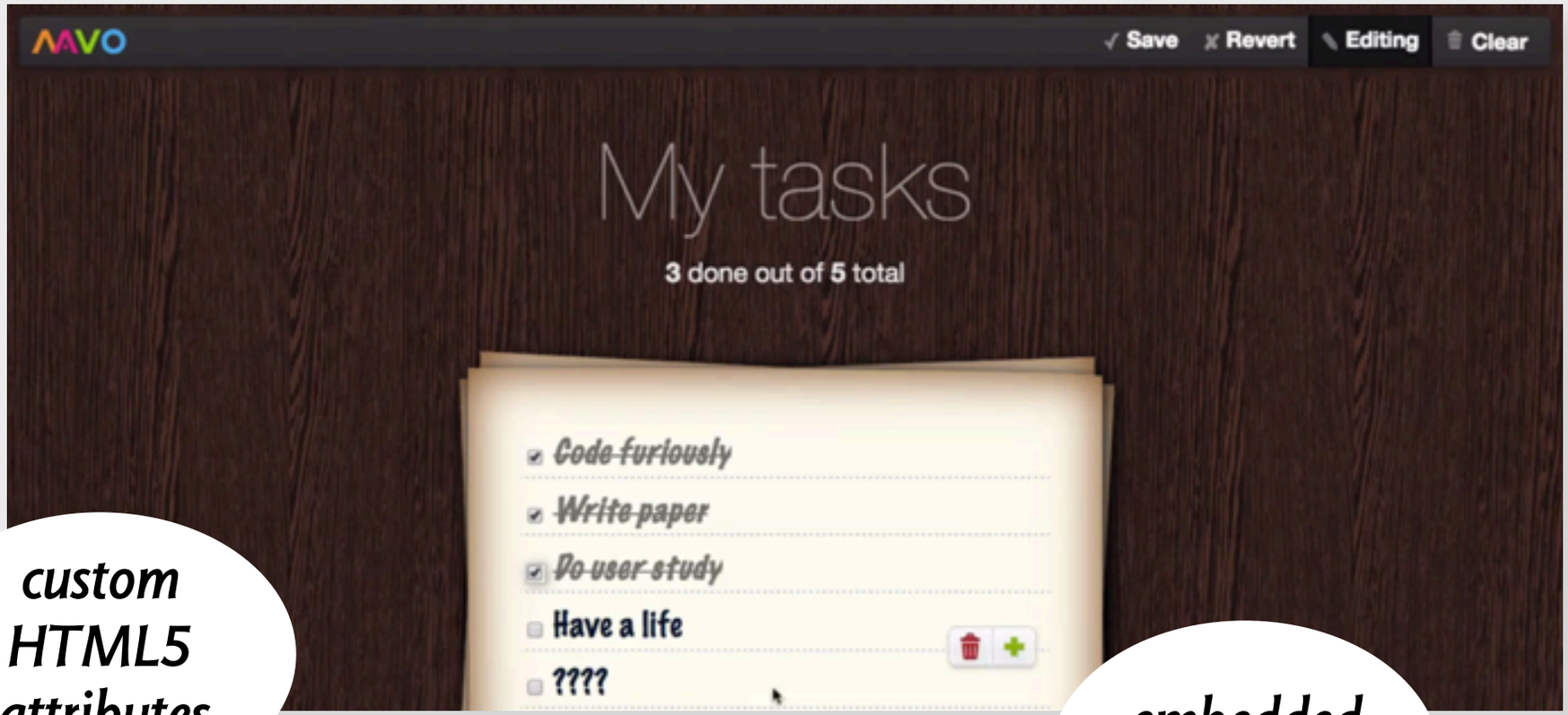
```
<body data-store="local">
    <h1>My tasks</h1>
    <p>[count(done)] done of [count(task)] total</p>

    <ul>
        <li property="task" data-multiple>
            <input property="done" type="checkbox" />
            <span property="taskTitle">Do stuff</span>
        </li>
    </ul>
</body>
```

data becomes schema

implicit editing controls

to-do in Mavo

the app



custom HTML5 attributes

embedded formula

the code

```
<body data-store="local">
    <h1>My tasks</h1>
    <p>[count(done)] done of [count(task)] total</p>

    <ul>
        <li property="task" data-multiple>
            <input property="done" type="checkbox" />
            <span property="taskTitle">Do stuff</span>
        </li>
    </ul>
</body>
```

data becomes schema

implicit editing controls

a new data store model

a new data store model

spreadsheet

a new data store model

spreadsheet

intuitive visual layout
schema evolves with data
can see all the data

a new data store model

spreadsheet

intuitive visual layout
schema evolves with data
can see all the data

numeric queries only
can't handle nested data
risky to insert/delete rows

a new data store model

spreadsheet

intuitive visual layout
schema evolves with data
can see all the data

numeric queries only
can't handle nested data
risky to insert/delete rows

relational database

a new data store model

spreadsheet

intuitive visual layout
schema evolves with data
can see all the data

numeric queries only
can't handle nested data
risky to insert/delete rows

relational database

rich query language
can encode structured data
easy to insert/delete tuples

a new data store model

spreadsheet

intuitive visual layout
schema evolves with data
can see all the data

numeric queries only
can't handle nested data
risky to insert/delete rows

relational database

rich query language
can encode structured data
easy to insert/delete tuples

not intuitive to end users
hard to evolve schema
seeing data needs queries

a new data store model

spreadsheet

intuitive visual layout
schema evolves with data
can see all the data

numeric queries only
can't handle nested data
risky to insert/delete rows

relational database

rich query language
can encode structured data
easy to insert/delete tuples

not intuitive to end users
hard to evolve schema
seeing data needs queries

a new data store model

spreadsheet	relational database
intuitive visual layout schema evolves with data can see all the data	rich query language can encode structured data easy to insert/delete tuples
numeric queries only can't handle nested data risky to insert/delete rows	not intuitive to end users hard to evolve schema seeing data needs queries

object sheets

intuitive visual layout
schema evolves with data
can see all the data

rich query language
can encode structured data
easy to insert/delete tuples

challenges
a new data model
a new query language
connection to clients

example: allocating offices

Room	Sq. footage	Occupant	Role
Dungeon Five	480	Sirius	Grad. student
		James	Post-doc
		Wormtail	Grad. student
Greenhouse Two	561	Bellatrix	Visiting Prof.
		Lily	Post-doc
		Remus	Post-doc

Role	Allocated space
Grad. student	12
Post-doc	20
Visiting Prof.	45

example: allocating offices

Room	Sq. footage	Occupant	Role
Dungeon Five	480	Sirius	Grad. student
		James	Post-doc
		Wormtail	Grad. student
Greenhouse Two	561	Bellatrix	Visiting Prof.
		Lily	Post-doc
		Remus	Post-doc

nested objects

Role	Allocated space
Grad. student	12
Post-doc	20
Visiting Prof.	45

example: allocating offices

Room	Sq. footage	Occupant	Role
Dungeon Five	480	Sirius	Grad. student
		James	Post-doc
		Wormtail	Grad. student
Greenhouse Two	561	Bellatrix	Visiting Prof.
		Lily	Post-doc
		Remus	Post-doc

nested objects

object references

Role	Allocated space
Grad. student	12
Post-doc	20
Visiting Prof.	45

in google spreadsheet

	A	B	C	D	E	F
1	room	sq. footage	occupant	role	alloc.	free
2	Dungeon Five	480	Sirius	Grad. student	12	436
3			James	Post-doc	20	
4			Wormtail	Grad. student	12	
5			Harry	Grad. student	12	
6	Greenhouse Two	561	Bellatrix	Visiting Prof.	45	476
7			Lily	Post-doc	20	
8			Remus	Post-doc	20	
9	role	alloc. space				
10	Grad. student	12			=VLOOKUP(D2, Sheet1!A10:B12,	
11	Post-doc	20			=B2 - SUM(E2:E4)	
12	Visiting Prof.	45			=B6 - SUM(E6:E8)	
13						

in google spreadsheet

but nesting is only visual, not computational

	A	B	C	D	E	F
1	room	sq. footage	occupant		c.	free
2	Dungeon Five	480	Sirius		12	436
3			James	Post-doc	20	
4			Wormtail	Grad. student	12	
5			Harry	Grad. student	12	
6	Greenhouse Two	561	Bellatrix	Visiting Prof.	45	476
7			Lily	Post-doc	20	
8			Remus	Post-doc	20	
9	role	alloc. space				
10	Grad. student	12			=VLOOKUP(D2, Sheet1!A10:B12,	
11	Post-doc	20			=B2 - SUM(E2:E4)	
12	Visiting Prof.	45			=B6 - SUM(E6:E8)	
13						

in google spreadsheet

	A	B	C	D	E	F
1	room	sq. footage	occupant		c.	free
2	Dungeon Five	480	Sirius		12	436
3			James	Post-doc	20	
4			Wormtail	Grad. student	12	
5			Harry	Grad. student	12	
6	Greenhouse Two	561	Bellatrix	Visiting Prof.	45	476
7			Lily	Post-doc	20	
8			Remus	Post-doc	20	
9	role	alloc. space				
10	Grad. student	12			=VLOOKUP(D2, Sheet1!A10:B12,	
11	Post-doc	20			=B2 - SUM(E2:E4)	
12	Visiting Prof.	45			=B6 - SUM(E6:E8)	
13						

but nesting is only visual, not computational

formulas are complex

in google spreadsheet

	A	B	C	D	E	F
1	room	sq. footage	occupant		c.	free
2	Dungeon Five	480	Sirius		12	436
3			James	Post-doc	20	
4			Wormtail	Grad. student	12	
5			Harry	Grad. student	12	
6	Greenhouse Two	561	Bellatrix	Visiting Prof.	45	476
7			Lily	Post-doc	20	
8			Remus	Post-doc	20	
9	role	alloc. space				
10	Grad. student	12			=VLOOKUP(D2, Sheet1!A10:B12,	
11	Post-doc	20			=B2 - SUM(E2:E4)	
12	Visiting Prof.	45			=B6 - SUM(E6:E8)	
13						

but nesting is only visual, not computational

formulas are complex

formulas are unstable

in object sheets

Room

	name	sqFoot
•	text	number
•	Dungeon Five	480
•	Greenhouse Two	561

Occupant

	name	role	free
•	text	Role	number
•	Sirius	Grad. student	436
•	James	Post-doc	
•	Wormtail	Grad. student	
•	Bellatrix	Visiting Prof.	476
•	Lily	Post-doc	
•	Remus	Post-doc	

Role

	title	allocSpace
•	text	number
•	Grad. student	12
•	Post-doc	20
•	Visiting Prof.	45

in object sheets

Room

	name	sqFoot
•	Dungeon Five	480
•	Greenhouse Two	561

Occupant

	name	role	free
•	Sirius	Grad. student	436
•	James	Post-doc	
•	Wormtail	Grad. student	
•	Bellatrix	Visiting Prof.	476
•	Lily	Post-doc	
•	Remus	Post-doc	

Role

	title	allocSpace
•	Grad. student	12
•	Post-doc	20
•	Visiting Prof.	45

nesting is now semantic, not just visual

in object sheets

Room

	name	sqFoot
•	text	number
•	Dungeon Five	480
•	Greenhouse Two	561

Occupant

	name	role	free
•	text	Role	number
•	Sirius	Grad. student	436
•	James	Post-doc	
•	Wormtail	Grad. student	
•	Bellatrix	Visiting Prof.	476
•	Lily	Post-doc	
•	Remus	Post-doc	

Role

	title	allocSpace
•	text	number
•	Grad. student	12
•	Post-doc	20
•	Visiting Prof.	45

nesting is now semantic, not just visual

first-class object references

in object sheets

sqFoot - sum[o : Occupant](o.role.allocSpace)

Room

	name	sqFoot	free
•	text	number	number
•	Dungeon Five	480	436
•	Greenhouse Two	561	476

Occupant

	name	role
•	text	Role
•	Sirius	Grad. student
•	James	Post-doc
•	Wormtail	Grad. student
•	Bellatrix	Visiting Prof.
•	Lily	Post-doc
•	Remus	Post-doc

Role

	title	allocSpace
•	text	number
•	Grad. student	12
•	Post-doc	20
•	Visiting Prof.	45

nesting is now semantic, not just visual

first-class object references

in object sheets

formulas over sets (now stable)

```
sqFoot - sum[ o : Occupant ]( o.role.allocSpace )
```

Room

	name	sqFoot
•	text	number
•	Dungeon Five	480
•	Greenhouse Two	561

Occupant

	name	role	free
•	text	Role	number
•	Sirius	Grad. student	436
•	James	Post-doc	
•	Wormtail	Grad. student	
•	Bellatrix	Visiting Prof.	476
•	Lily	Post-doc	
•	Remus	Post-doc	

Role

	title	allocSpace
•	text	number
•	Grad. student	12
•	Post-doc	20
•	Visiting Prof.	45

nesting is now semantic, not just visual

first-class object references

a parent-teacher app

MAVO ✓ Save ✗ Revert

Back to login

Parent view for Molly

- Ronald
  - Potions with Snape: 2014-12-16 13:45
  - Defence with Snape:
- Ginevra
  - Potions with Snape:
  - Charms with Flitwick: 2014-12-17 13:00

what parent sees

MAVO ✎ Editing ✓ Save ✗ Revert 🗑 Clear

Back to login

Teacher view for Flitwick

Slot time	Scheduled meeting
2014-12-17 13:00	Ginevra in 6.005
2014-12-17 14:00	

Add slot

what teacher sees

MAVO ✎ Edit ✓ Save ✗ Revert 🗑 Clear

	2014-12-16	2014-12-17
13:00	Augustus / Snape	Molly / Flitwick
13:15		
13:30		
13:45	Molly / Snape	
14:00		

what principal sees

the backend

Person			Slot			
name	roles	parents	time	scheduledEnrollment	valid	
text	text	Person	text	Enrollment	bool	
Snape	teacher		2014-12-16 13:00	Seamus in 6.820	true	
			2014-12-16 13:30		true	
			2014-12-16 13:45	Ronald in 6.170	true	
Flitwick	teacher		2014-12-16 13:00	Ginevra in 6.005	true	
			2014-12-16 14:00		true	
Ronald	student	Molly				
Ginevra	student	Molly				
Seamus	student	Augustus				
Molly						
Augustus						

Class		Section	Enrollment		
code	name	teacher	student	scheduledSlot	valid
text	text	Person	Person	Slot	bool
6.170	Potions	Snape	Ronald	Snape @ 2014-12-16 13:45	true
			Ginevra		true
6.820	Defence	Snape	Ronald		true
			Seamus	Snape @ 2014-12-16 13:00	true
6.005	Charms	Flitwick	Ginevra	Flitwick @ 2014-12-16 13:00	true
			Seamus		true

the backend

Person			Slot			
name	roles	parents	time	scheduledEnrollment	valid	
text	text	Person	text	Enrollment	bool	
Snape	teacher		2014-12-16 13:00	Seamus in 6.820	true	
			2014-12-16 13:30		true	
			2014-12-16 13:45	Ronald in 6.170	true	
Flitwick	teacher		2014-12-16 13:00	Ginevra in 6.005	true	
			2014-12-16 14:00		true	
Ronald	student	Molly				
Ginevra	student	Molly				
Seamus	student	Augustus				
Molly						
Augustus						

Class		Section	Enrollment		
code	name	teacher	student	scheduledSlot	valid
text	text	Person	Person	Slot	bool
6.170	Potions	Snape	Ronald	Snape @ 2014-12-16 13:45	true
			Ginevra		true
6.820	Defence	Snape	Ronald		true
			Seamus	Snape @ 2014-12-16 13:00	true
6.005	Charms	Flitwick	Ginevra	Flitwick @ 2014-12-16 13:00	true
			Seamus		true

```
scheduledEnrollment.Section.teacher = Person
```

"the teacher of the section of this enrollment
is the person for this slot"

code reductions for some apps

	Original Implementation			Objsheets+Mavo		
Application	Language/Framework	Code (LoC)	HTML (LoC)	Formulas (count)	Macros (LoC)	HTML (LoC)
PTC	(unavailable to us)			20	29	73
TodoMVC	JavaScript/Angular	113	75	6	5	22
Conf	Python/Django	694	399	30	62	154
HackQ	JavaScript/Meteor	158	142	2	13	110
Got Milk	Perl/CGI	188	40	2	15	26

conclusions

pros and cons

pros and cons

bringing together
visual GUI building
model-driven dev
declarative queries

mature platforms
especially larger players

integration
web service APIs
enterprise backends

easy ramp-up
simple things are simple
but gets harder fast

pros and cons

bringing together
visual GUI building
model-driven dev
declarative queries

mature platforms
especially larger players

integration
web service APIs
enterprise backends

easy ramp-up
simple things are simple
but gets harder fast

technology limitations
ad hoc limitations
query language ≪ SQL
non-declarative scripts

sidesteps standard tools
automated testing
version control
even collaboration

talent shortage
may be hard to hire
few resources online

stackoverflow tags for:

rails	272674
spring	108989
excel-vba	60557
meteor	25549
sap	3356
abap	1316
servicenow	309
nintex	125
k2	115
outsystems	80
quickbase	58
mendix	8
bizagi	8
appian	4
salesforce app cloud	4
agilepoint	2
caspio	0
matssoft	0
micropact	0
miosoft	0

questions for discussion

questions for discussion

diagrams better than text?
what about sharing?

questions for discussion

diagrams better than text?
what about sharing?

what class of apps are low code platforms suited to?
"community apps"? enterprise CRUD apps?

questions for discussion

diagrams better than text?
what about sharing?

what class of apps are low code platforms suited to?
"community apps"? enterprise CRUD apps?

what's the impact on shadow IT?
opportunity for coordinated citizen development?

questions for discussion

diagrams better than text?
what about sharing?

what class of apps are low code platforms suited to?
"community apps"? enterprise CRUD apps?

what's the impact on shadow IT?
opportunity for coordinated citizen development?

slides, papers, links at: **tiny.cc/lowcode**