Functional Reactive Programming
Another approach to asynchronous systems

Federico Builes
federico@mheroin.com
Framework funcional declarativo para el procesamiento streams de eventos
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: function(data, status, jqXhr){
        loader.stop();
        showLyrics();
        $("#set-video").show();
        if (data === "Sorry, We don't have lyrics for this song yet.") {
            activateStep("step2-nolyrics");
        } else {
            $("#fetch-lyrics").hide();
            activateStep("step2");
        }
        write(data);
    },
    error: function(xhr, status, error){
        loader.stop();
        showLyrics();
        if (artist === "" || title === "") {
            write("You need to enter the artist's name and the song title.");
        } else {
            write("There has been an error processing the data. Please try again.");
        }
    }
});
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: function(data, status, jqXhr){
        loader.stop();
        showLyrics();
        $('#set-video').show();
        if (data === "Sorry, We don't have lyrics for this song yet.") {
            activateStep("step2-nolyrics");
        } else {
            $('#fetch-lyrics').hide();
            activateStep("step2");
        }
        write(data);
    },
    error: function(xhr, status, error){
        loader.stop();
        showLyrics();
        if (artist === "" || title === "") {
            write("You need to enter the artist's name and the song title.");
        } else {
            write("There has been an error processing the data. Please try again.");
        }
    }
});
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: function(data, status, jqXhr){
        loader.stop();
        showLyrics();
        $('#set-video').show();
        if (data === "Sorry, We don't have lyrics for this song yet.") {
            activateStep("step2-nolyrics");
        } else {
            $('#fetch-lyrics').hide();
            activateStep("step2");
        }
        write(data);
    },
    error: function(xhr, status, error){
        loader.stop();
        showLyrics();
        if (artist === "" || title === ") {
            write("You need to enter the artist's name and the song title.");
        } else {
            write("There has been an error processing the data. Please try again.");
        }
    }
});
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: ajaxSuccess,
    error: ajaxError,
});
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: ajaxSuccess,
    error: ajaxError,
});
Continuation-passing Style (CPS)
Continuation-passing style (CPS) is a style of programming in which control is passed explicitly in the form of a continuation.
A continuation is a data structure that represents the computational process at a given point in the process’ execution.
function id(x) {
    return x;
}

// CPS

function id(x, cc) {
    cc(x);
}
function fact(n) {
    if (n == 0)
        return 1;
    else
        return n * fact(n-1);
}

// En CPS
function fact(n, cc) {
    if (n == 0)
        cc(1);
    else {
        fact(n - 1, function (val) {
            cc(n * val)
        });
    }
}
fact (5, function (n) {
    console.log(n);
});
(defun pyth (x y) (sqrt (+ (* x x) (* y y))))
(defun cps+ (x y cc)  
  (funcall cc (+ x y)))

(defun cps* (x y cc)  
  (funcall cc (* x y)))

(defun sqrt* (x cc)  
  (funcall cc (sqrt x)))

(defun cps-pyth (x y cc)  
  (cps* x x (lambda (x2)  
    (cps* y y (lambda (y2)  
      (cps+ x2 y2 (lambda (sum)  
        (sqrt* sum cc))))))))

viva Lisp =)
add_cps :: Int -> Int -> (Int -> r) -> r
add_cps x y k = k (x + y)

square_cps :: Int -> (Int -> r) -> r
square_cps x k = k (x * x)

pythagoras_cps :: Int -> Int -> (Int -> r) -> r
pythagoras_cps x y k =
  square_cps x $ \x_squared \rightarrow
  square_cps y $ \y_squared \rightarrow
  add_cps x_squared y_squared $ \sum_of_squares \rightarrow
  k \sum_of_squares
add_cps :: Int -> Int -> (Int -> r) -> r
add_cps x y k = k (x + y)

square_cps :: Int -> (Int -> r) -> r
square_cps x k = k (x * x)

pythagoras_cps :: Int -> Int -> (Int -> r) -> r
pythagoras_cps x y k =
  square_cps x $ \x_squared ->
  square_cps y $ \y_squared ->
  add_cps x_squared y_squared $ \sum_of_squares ->
k sum_of_squares
Compiladores/Analizadores
Cambios en Flujo
Concurrencia
Sistemas Distribuidos
Sistemas con Latencia Inherente
Callback Hell
$.ajax(
    {
        url: apiUrl,
        method: apiMethod,
        data: { artist: artist, title: title },
        beforeSend: loader.start,
        success: function(data, status, jqXhr){
            loader.stop();
            showLyrics();
            $("#set-video").show();
            if (data === "Sorry, We don't have lyrics for this song yet.") {
                activateStep("step2-nolylrics");
            } else {
                $("#fetch-lyrics").hide();
                activateStep("step2");
            }
            write(data);
        },
        error: function(xhr, status, error){
            loader.stop();
            showLyrics();
            if (artist === "" || title === "") {
                write("You need to enter the artist's name and the song title.");
            } else {
                write("There has been an error processing the data. Please try again.");
            }
        }
    });
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: ajaxSuccess,
    error: ajaxError,
});
$.ajax({
    url: apiUrl,
    method: apiMethod,
    data: { artist: artist, title: title },
    beforeSend: loader.start,
    success: ajaxSuccess,
    error: ajaxError,
});
$ find . -name *.js | xargs cat | wc -l
  1019

$ ack "\.ajax\(\{\"" | wc -l
    48
goto : structured programming

callbacks : async programming
Functional Reactive Programming
Functional Reactive Programming
<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$15,000</td>
<td>Vehicle Price</td>
<td>$10,750</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>227.5%</td>
</tr>
<tr>
<td>3</td>
<td>$14,750</td>
<td>$11,000</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>164.3%</td>
<td>$6.61</td>
</tr>
<tr>
<td>4</td>
<td>$14,500</td>
<td>$11,250</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>112.6%</td>
<td>$5.32</td>
</tr>
<tr>
<td>5</td>
<td>$14,250</td>
<td>$11,500</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>71.9%</td>
<td>$4.30</td>
</tr>
<tr>
<td>6</td>
<td>$14,000</td>
<td>$11,750</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>41.5%</td>
<td>$3.54</td>
</tr>
<tr>
<td>7</td>
<td>$13,750</td>
<td>$12,000</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>20.6%</td>
<td>$3.02</td>
</tr>
<tr>
<td>8</td>
<td>$13,500</td>
<td>$12,250</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>7.9%</td>
<td>$2.70</td>
</tr>
<tr>
<td>9</td>
<td>$13,250</td>
<td>$12,500</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>1.8%</td>
<td>$2.55</td>
</tr>
<tr>
<td>10</td>
<td>$13,000</td>
<td>$12,750</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>0.1%</td>
<td>$2.50</td>
</tr>
<tr>
<td>11</td>
<td>$12,750</td>
<td>$13,000</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-0.1%</td>
<td>$2.50</td>
</tr>
<tr>
<td>12</td>
<td>$12,500</td>
<td>$13,250</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-2.2%</td>
<td>$2.45</td>
</tr>
<tr>
<td>13</td>
<td>$12,250</td>
<td>$13,500</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-10.6%</td>
<td>$2.23</td>
</tr>
<tr>
<td>14</td>
<td>$12,000</td>
<td>$13,750</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-31.0%</td>
<td>$1.72</td>
</tr>
<tr>
<td>15</td>
<td>$11,750</td>
<td>$14,000</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-70.2%</td>
<td>$1.00</td>
</tr>
<tr>
<td>16</td>
<td>$11,500</td>
<td>$14,250</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-136.7%</td>
<td>$1.00</td>
</tr>
<tr>
<td>17</td>
<td>$11,250</td>
<td>$14,500</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-241.1%</td>
<td>$1.00</td>
</tr>
<tr>
<td>18</td>
<td>$11,000</td>
<td>$14,750</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-396.2%</td>
<td>$1.00</td>
</tr>
<tr>
<td>19</td>
<td>$10,750</td>
<td>$15,000</td>
<td>100.0%</td>
<td>100.0%</td>
<td>$2.50</td>
<td>$1.00</td>
<td>$9.00</td>
<td>-617.9%</td>
<td>$1.00</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Rows 20 and beyond are not shown.
Functional
Reactive
Programming
Behaviors

(signals)

Valores reactivos que varían en el tiempo
Behavior $a = \text{Time} \rightarrow a$

El valor del behavior $s$ en el tiempo $t$ es $s(t)$
Event

Sencuencias de ocurrencias de eventos en el tiempo
Event a = Time × a

Pares: (tiempo, información sobre la ocurrencia evento)
leftClick :: Event ()
keyPress :: Event Char
Behaviors Constantes

red :: Behavior Color
1 :: Behavior Real
Behaviors Variables

time :: Behavior Time*
-=> :: Event\(\alpha\) \rightarrow \beta \rightarrow \text{Event}\(\beta\)

color :: Behavior Color
color = red `until` (leftClick \rightarrow blue)

circ :: Behavior Region
circ = translate (cos time, sin time) (circle 1)

ball :: Behavior Picture
ball = paint color circ
Barack Obama
@BarackObama
This account is run by #Obama2012 campaign staff. Tweets from the President are signed -bo.
Washington, DC · http://www.barackobama.com

7,933 TWEETS 670,662 FOLLOWING 22,980,435 FOLLOWERS

Tweets

Barack Obama @BarackObama
The President made a surprise visit to campaign HQ yesterday to thank staff & volunteers. Here's what he said. OFA.BO/Bu2y3i

Barack Obama @BarackObama
The definition of hope is you still believe, even when it's hard. pic.twitter.com/BJCKP2aT
def follow
    res = write_to_db
    res = notify_user(res)
    return update_ui(res)
end

def write_to_db
    # ...
end

def notify_user
    # ...
end

def update_ui
    # ...
end
def follow
    write_to_db.callback do |a|
        notify_user(a).callback do |b|
            update_ui(b).callback do |resp|
                puts resp
            end
        end
    end
end
# follow es el evento
def follow
  tie(:write_to_db, :notify_user, :update_ui)
end
Referencias


• http://elm-lang.org/learn/What-is-FRP.elm

• http://en.wikibooks.org/wiki/Haskell/Continuation_passing_style