Conquering Callback Mountain

Sane non-blocking code with
Mojo::IOLoop::Delay

Follow along!

github.com/kanatohodets/delays-talk

Where's the (block) party?

- Ask Willem Alexander
  - Museumplein 1m30s
- Ask Cees
  - Rembrandtplein 45s
- Ask Jos
  - Jos' Awesome Partyfeest 2m15s
- Time to party (TTP): **4m30s**

Where’s the party?

- Ask Willem Alexander
- Ask Cees
- Ask Jos

• Cees: Rembrandtplein	45s
• WA: Museumplein	1m30s
• Jos: Jos’ awesome partyfeest	2m15s

TTP?

2m 15s!

Amazing Non-blockage!

- Faster response times
- More efficient resource usage
- Hip / trendy / used by rockstar ninjas

WEBSCALE

How do I async?

Aside: Mojo

```
#!perl
use v5.20.1; use warnings;
use Mojolicious::Lite;
use Mojo::Pg;
use Mojo::IOLoop;
use experimental qw(signatures postderef);

helper pg =>
  sub { state $pg = Mojo::Pg->new('postgresql://
btyler@localhost/my_cool_db') };
#… stuff
app->start;
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

How do I async?

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

Argh!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

Argh!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

Argh!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

EVERYTHING IS
TERRIBLE!

Delays!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $d->data(name => $name);
            $c->pg->db->query('SELECT id FROM customers WHERE name = ?', $name => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $customer_id = $res->array->[0];

            $d->data(customer_id => $customer_id);
            $c->pg->db->query('SELECT COUNT(1) FROM beer_log WHERE customer_id = ?', $customer_id => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $count = $res->array->[0];

            $d->data(count => $count);
            $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start', $count => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $price = $res->array->[0];

            my ($name, $customer_id, $count) = $d->data->@{qw(name customer_id count)};
            $c->render(json => { name => $name, id => $customer_id, beers => $count, price => $price });
        })->catch(sub ($d, $err) {
            $c->render(text => "blorg error! $err", code => 500);
    })->wait;
};
```

Delays!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $d->data(name => $name);
            $c->pg->db->query('SELECT id FROM customers WHERE name = ?', $name => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $customer_id = $res->array->[0];

            $d->data(customer_id => $customer_id);
            $c->pg->db->query('SELECT COUNT(1) FROM beer_log WHERE customer_id = ?', $customer_id => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $count = $res->array->[0];

            $d->data(count => $count);
            $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start', $count => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $price = $res->array->[0];

            my ($name, $customer_id, $count) = $d->data->@{qw(name customer_id count)};
            $c->render(json => { name => $name, id => $customer_id, beers => $count, price => $price });
        })->catch(sub ($d, $err) {
            $c->render(text => "blorg error! $err", code => 500);
    })->wait;
};
```

Delays!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;
    # … stuff
};
```

Delays!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $d->data(name => $name);
            $c->pg->db->query('SELECT id FROM customers WHERE name = ?', $name => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $customer_id = $res->array->[0];

            $d->data(customer_id => $customer_id);
            $c->pg->db->query('SELECT COUNT(1) FROM beer_log WHERE customer_id = ?', $customer_id => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $count = $res->array->[0];

            $d->data(count => $count);
            $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start', $count => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $price = $res->array->[0];

            my ($name, $customer_id, $count) = $d->data->@{qw(name customer_id count)};
            $c->render(json => { name => $name, id => $customer_id, beers => $count, price => $price });
        })->catch(sub ($d, $err) {
            $c->render(text => "blorg error! $err", code => 500);
    })->wait;
};
```

Structure

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->wait;
```

Structure

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->wait;
```

Structure

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->wait;
```

Structure

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->wait;
```

Structure

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->wait;
```

Structure

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->wait;
```

Structure (plus errors)

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->catch(sub ($d, $err) {
    #blorg, an error!
})->wait;
```

Structure (plus errors)

```
# … stuff
my $delay = Mojo::IOLoop->delay(
    sub ($d) {  }, # step 1
    sub ($d, @args) {  }, # step 2
    sub ($d, @args) {  }, # step 3
    sub ($d, @args) {  } # finish
)->catch(sub ($d, $err) {
    #blorg, an error!
})->wait;
```

Delays!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $d->data(name => $name);
            $c->pg->db->query('SELECT id FROM customers WHERE name = ?', $name => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $customer_id = $res->array->[0];

            $d->data(customer_id => $customer_id);
            $c->pg->db->query('SELECT COUNT(1) FROM beer_log WHERE customer_id = ?', $customer_id => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $count = $res->array->[0];

            $d->data(count => $count);
            $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start', $count => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $price = $res->array->[0];

            my ($name, $customer_id, $count) = $d->data->@{qw(name customer_id count)};
            $c->render(json => { name => $name, id => $customer_id, beers => $count, price => $price });
        })->catch(sub ($d, $err) {
            $c->render(text => "blorg error! $err", code => 500);
    })->wait;
};
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

First Steps

```
sub ($d) {
    $d->data(name => $name);
    $c->pg->db->query('SELECT id FROM customers
WHERE name = ?', $name => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $customer_id = $res->array->[0];

    $d->data(customer_id => $customer_id);
    $c->pg->db->query('SELECT COUNT(1) FROM beer_log
WHERE customer_id = ?', $customer_id => $d->begin);
},
# … more steps in the delay
```

Delays!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $d->data(name => $name);
            $c->pg->db->query('SELECT id FROM customers WHERE name = ?', $name => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $customer_id = $res->array->[0];

            $d->data(customer_id => $customer_id);
            $c->pg->db->query('SELECT COUNT(1) FROM beer_log WHERE customer_id = ?', $customer_id => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $count = $res->array->[0];

            $d->data(count => $count);
            $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start', $count => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $price = $res->array->[0];

            my ($name, $customer_id, $count) = $d->data->@{qw(name customer_id count)};
            $c->render(json => { name => $name, id => $customer_id, beers => $count, price => $price });
        })->catch(sub ($d, $err) {
            $c->render(text => "blorg error! $err", code => 500);
    })->wait;
};
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Final Steps

```
# … previous delay steps
sub ($d, $err, $res) {
    die $err if $err;
    my $count = $res->array->[0];

    $d->data(count => $count);
    $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ?
>= range_start', $count => $d->begin);
},
sub ($d, $err, $res) {
    die $err if $err;
    my $price = $res->array->[0];

    my ($name, $customer_id, $count) =
        $d->data->@{qw(name customer_id count)};
    $c->render(json => { name => $name, id => $customer_id, beers =>
$count, price => $price });
}
# … catch and ->wait
```

Ta-da!

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $d->data(name => $name);
            $c->pg->db->query('SELECT id FROM customers WHERE name = ?', $name => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $customer_id = $res->array->[0];

            $d->data(customer_id => $customer_id);
            $c->pg->db->query('SELECT COUNT(1) FROM beer_log WHERE customer_id = ?', $customer_id => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $count = $res->array->[0];

            $d->data(count => $count);
            $c->pg->db->query('SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start', $count => $d->begin);
        },
        sub ($d, $err, $res) {
            die $err if $err;
            my $price = $res->array->[0];

            my ($name, $customer_id, $count) = $d->data->@{qw(name customer_id count)};
            $c->render(json => { name => $name, id => $customer_id, beers => $count, price => $price });
        })->catch(sub ($d, $err) {
            $c->render(text => "blorg error! $err", code => 500);
    })->wait;
};
```

Callbacks

```
get '/booze_check' => sub ($c) {
    my $name = $c->param('name');
    $c->render_later;

    my $customer_sql = 'SELECT id FROM customers WHERE name = ?';
    $c->pg->db->query($customer_sql, $name => sub ($db, $err, $res) {
        my $customer_id = $res->expand->hashes->[0]->{id};

        my $count_sql = 'SELECT COUNT(1) FROM beer_log WHERE customer_id = ?';
        $c->pg->db->query($count_sql, $customer_id => sub ($db, $err, $res) {
            my $count = $res->hashes->[0]->{count};

            my $price_sql = 'SELECT MAX(price) FROM beer_price_scale WHERE ? >= range_start';
            $c->pg->db->query($price_sql, $count => sub ($db, $err, $res) {
                my $price = $res->array->[0];

                $c->render(json => { name => $name, id => $customer_id,
                                     beers => $count, price => $price });
            });
        });
    });
};
```

Questions?

Bonus!

```
get '/multi_query' => sub ($c) {
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $c->pg->db->query('select ?::text, pg_sleep(4)',
                'Cees' => $d->begin);
            $c->pg->db->query('select ?::text, pg_sleep(2)',
                'Jos' => $d->begin);
        },
        sub ($d, $cees_err, $cees_res, $jos_err, $jos_res) {
            $c->render(json => {
                cees => $cees_res->hashes->[0],
                jos => $jos_res->hashes->[0]
            });
        }
    )->wait;
};
```

Bonus!

```
get '/multi_query' => sub ($c) {
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $c->pg->db->query('select ?::text, pg_sleep(4)',
                'Cees' => $d->begin);
            $c->pg->db->query('select ?::text, pg_sleep(2)',
                'Jos' => $d->begin);
        },
        sub ($d, $cees_err, $cees_res, $jos_err, $jos_res) {
            $c->render(json => {
                cees => $cees_res->hashes->[0],
                jos => $jos_res->hashes->[0]
            });
        }
    )->wait;
};
```

Bonus!

```
get '/multi_query' => sub ($c) {
    $c->render_later;

    my $delay = Mojo::IOLoop->delay(
        sub ($d) {
            $c->pg->db->query('select ?::text, pg_sleep(4)',
                'Cees' => $d->begin);
            $c->pg->db->query('select ?::text, pg_sleep(2)',
                'Jos' => $d->begin);
        },
        sub ($d, $cees_err, $cees_res, $jos_err, $jos_res) {
            $c->render(json => {
                cees => $cees_res->hashes->[0],
                jos => $jos_res->hashes->[0]
            });
        }
    )->wait;
};
```

The End