Arrows in Commercial Web Applications

Eric Fritz Jose Antony Tian Zhao

University of Wisconsin - Milwaukee

Research History

Directing JavaScript with Arrows
Khoo Yit Phang, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal.
5th Symposium on Dynamic Languages, 2009.

Research History

Directing JavaScript with Arrows
Khoo Yit Phang, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal.
5th Symposium on Dynamic Languages, 2009.

Inferring Types of Asynchronous Arrows in JavaScript
Eric Fritz and Tian Zhao.
Reactive and Event-based Languages and Systems, 2015.

Research History

Directing JavaScript with Arrows
Khoo Yit Phang, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal.
5th Symposium on Dynamic Languages, 2009.

Inferring Types of Asynchronous Arrows in JavaScript
Eric Fritz and Tian Zhao.
Reactive and Event-based Languages and Systems, 2015.

Typing and Semantics of Asynchronous Arrows in JavaScript
Eric Fritz and Tian Zhao.
In Review, 2016.

Research History

Directing JavaScript with Arrows
Khoo Yit Phang, Michael Hicks, Jeffrey S. Foster, and Vibha Sazawal.
5th Symposium on Dynamic Languages, 2009.

Inferring Types of Asynchronous Arrows in JavaScript
Eric Fritz and Tian Zhao.
Reactive and Event-based Languages and Systems, 2015.

Typing and Semantics of Asynchronous Arrows in JavaScript
Eric Fritz and Tian Zhao.
In Review, 2016.

Arrows in Commercial Web Applications
Eric Fritz, Jose Antony, and Tian Zhao.
HotWeb, 2016.

Example Application - Inventory Search

bookcase

Prev · Displaying 21-24 of 24 · Next

ID	Name	Category	Subcategry	Price / Unit	Margin
223	Rush Hierlooms Collection 1" Thick Stackable Bookcases	Furniture	Bookcases	170.98	0.66
224	Rush Hierlooms Collection Rich Wood Bookcases	Furniture	Bookcases	160.98	0.72
230	Safco Value Mate Steel Bookcase, Baked Enamel Finish on Steel, Black	Furniture	Bookcases	70.98	0
231	Sauder Camden County Barrister Bookcase, Planked Cherry Finish	Furniture	Bookcases	120.98	0.71

Example Application - Inventory Search

bookcase

Prev · Displaying 21-24 of 24 · Next

ID	Name	Category	Subcategry	Price / Unit	Margin
223	Rush Hierlooms Collection 1" Thick Stackable Bookcases	Furniture	Bookcases	170.98	0.66
224	Rush Hierlooms Collection Rich Wood Bookcases	Furniture	Bookcases	160.98	0.72
230	Safco Value Mate Steel Bookcase, Baked Enamel Finish on Steel, Black	Furniture	Bookcases	70.98	0
231	Sauder Camden County Barrister Bookcase, Planked Cherry Finish	Furniture	Bookcases	120.98	0.71

Requirements:

Filter displayed results by name and category

Paginate results when filtered set is $>$ 50 items

Example Application - Inventory Search

bookcase

Prev · Displaying 21-24 of 24 · Next

ID	Name	Category	Subcategry	Price / Unit	Margin
223	Rush Hierlooms Collection 1" Thick Stackable Bookcases	Furniture	Bookcases	170.98	0.66
224	Rush Hierlooms Collection Rich Wood Bookcases	Furniture	Bookcases	160.98	0.72
230	Safco Value Mate Steel Bookcase, Baked Enamel Finish on Steel, Black	Furniture	Bookcases	70.98	0
231	Sauder Camden County Barrister Bookcase, Planked Cherry Finish	Furniture	Bookcases	120.98	0.71

Requirements:

Filter displayed results by name and category

Paginate results when filtered set is $>$ 50 items

Performance Optimizations:

Cache server results for same query/page

Attempt to pre-fetch the next page of results

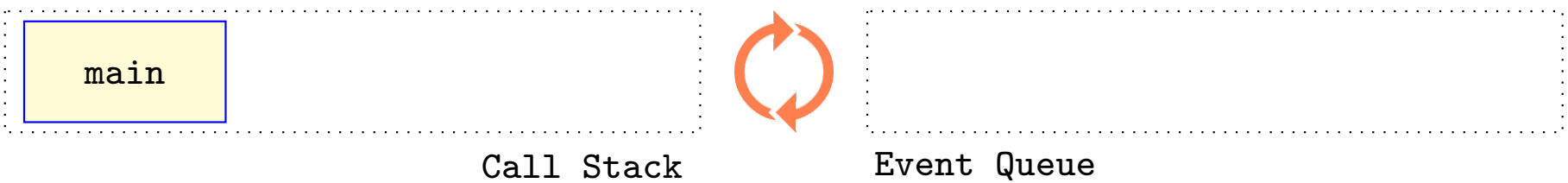
Don't make a remote request while the user is typing

Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```

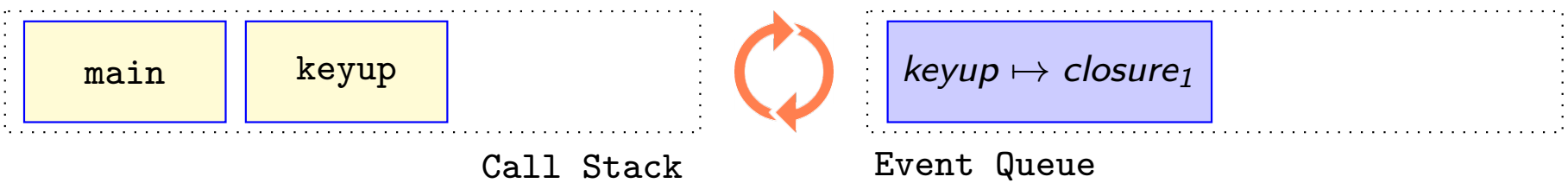
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



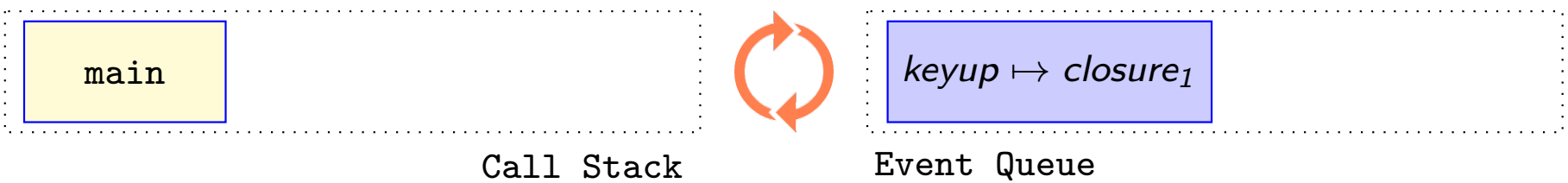
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



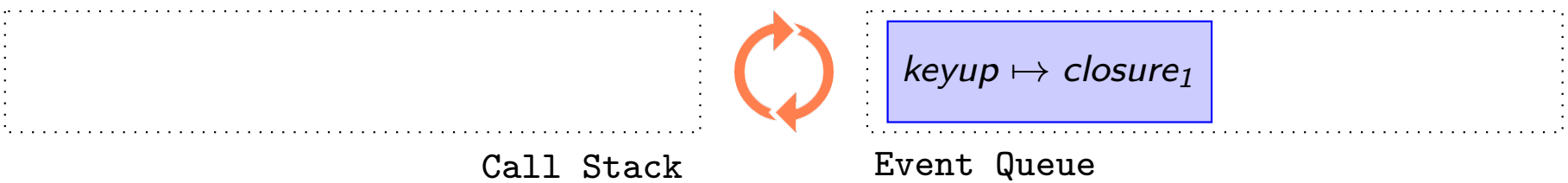
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



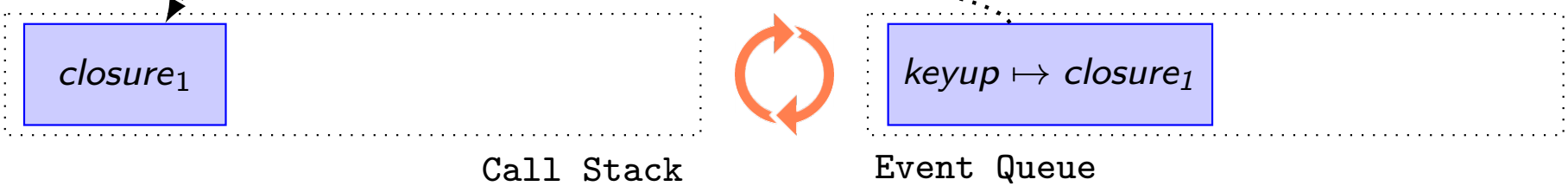
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



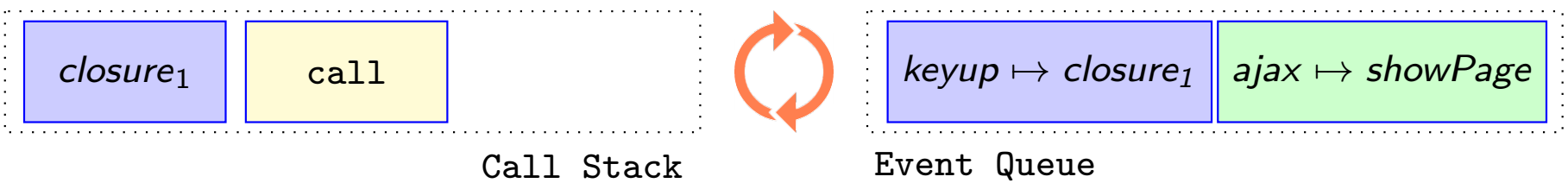
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



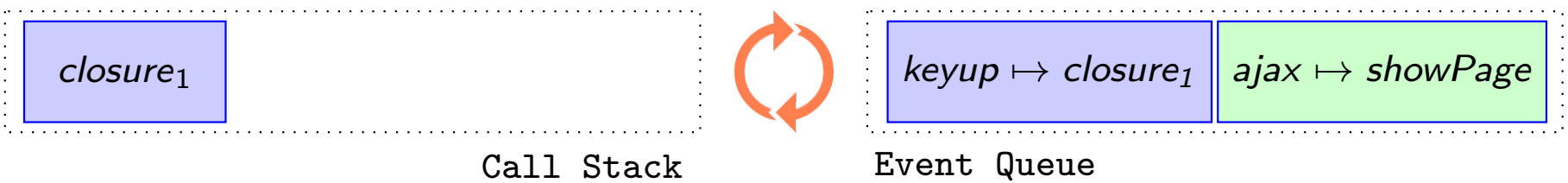
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



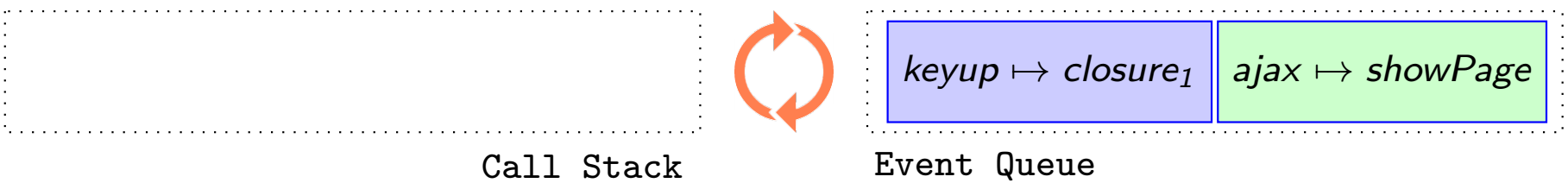
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



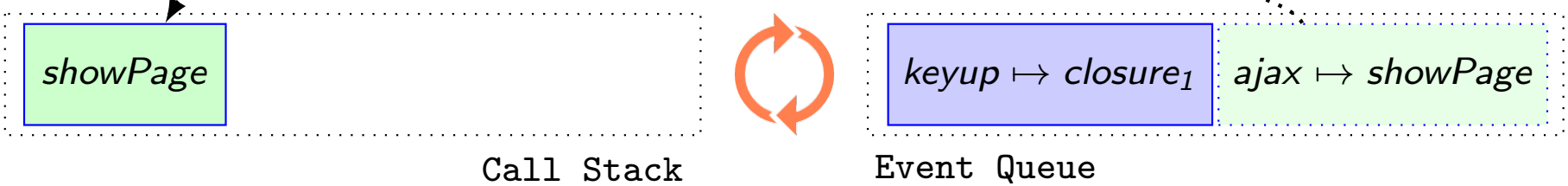
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



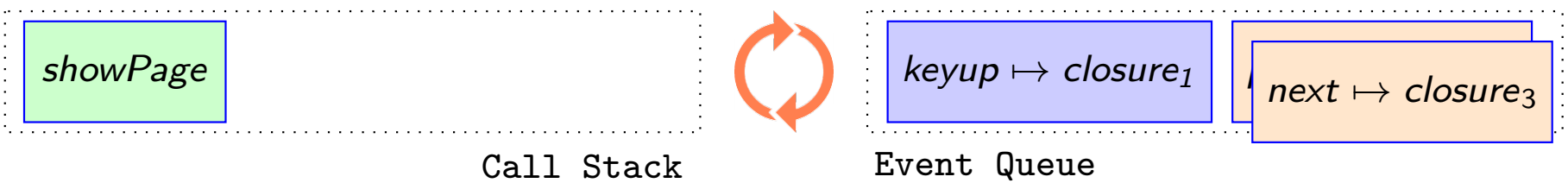
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



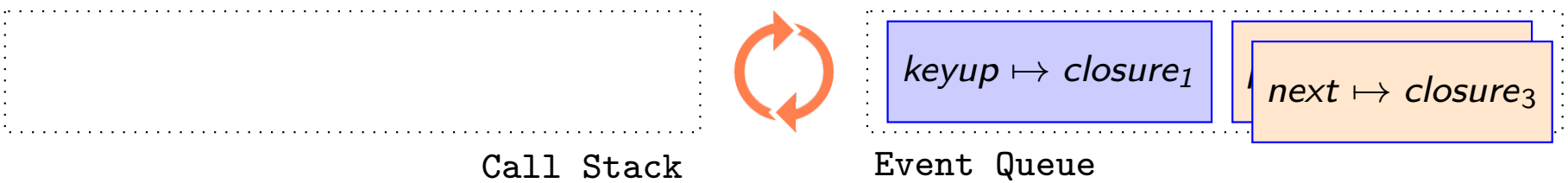
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



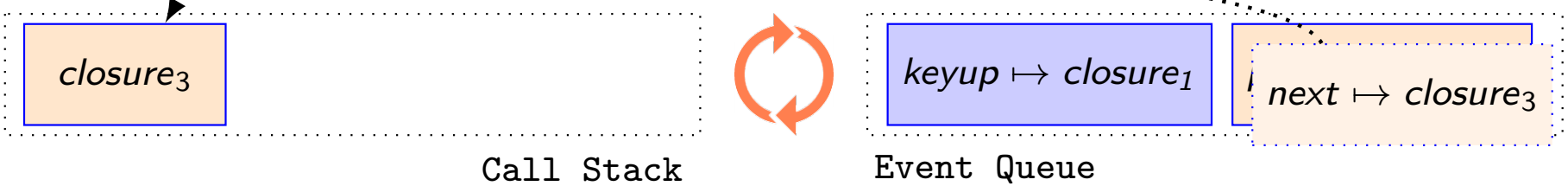
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



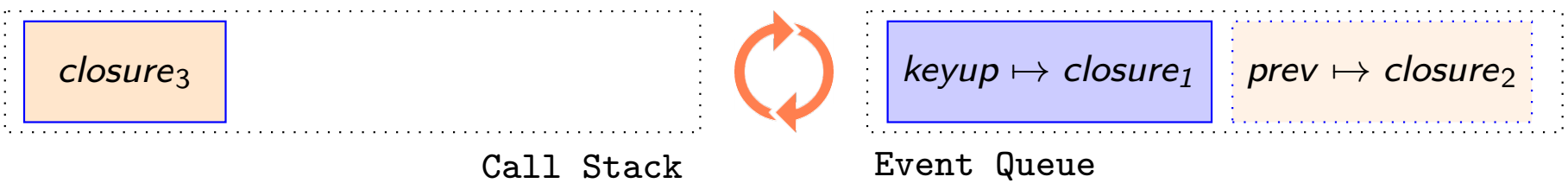
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



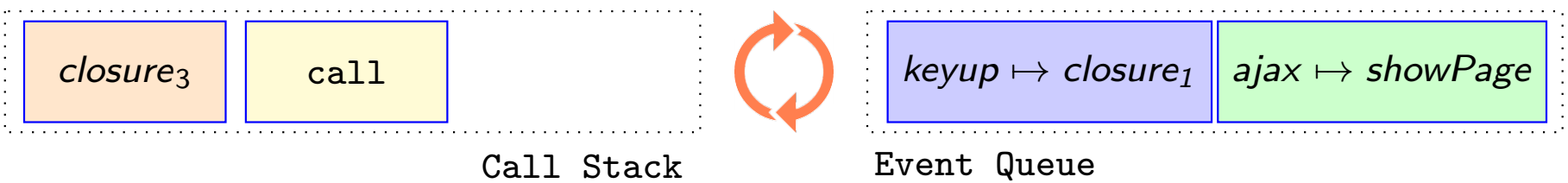
Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



Sequencing Filtering + Paging (Using Callbacks)

```
function call(query, page, handler) {
    $.ajax({ 'url': makeURL(query, page), 'success': handler });
}

function showPage(resp) {
    const pageTo = (page) => () => {
        $('#prev, #next').unbind('click');
        call(resp.query, page, showPage);
    };

    displayTable(resp);
    $('#prev').one('click', pageTo(resp.prev));
    $('#next').one('click', pageTo(resp.next));
}

$('#filter').keyup((ev) => {
    $('#prev, #next').unbind('click');
    call($(ev.target).val(), 1, showPage);
});
```



Callbacks - The Problems

Lose flow of (synchronous) code

Callbacks - The Problems

Lose flow of (synchronous) code

Event registration/de-registration is ad-hoc

Callbacks - The Problems

Lose flow of (synchronous) code

Event registration/de-registration is ad-hoc

Lose imperative exception semantics

```
try {
    User.findOne({name: 'Eric'}, (err, obj) => {
        if      (err != null) { throw err;                      }
        else if (obj == null) { throw new Error('Not Found'); }
        else                  { /*            ...            */ }
    });
} catch (err) {
    /* ... */
}
```

Callbacks - The Problems

Lose flow of (synchronous) code

Event registration/de-registration is ad-hoc

Lose imperative exception semantics

```
try {
    User.findOne({name: 'Eric'}, (err, obj) => {
        if       (err != null) { throw err;                       }
        else if (obj == null) { throw new Error('Not Found'); }
        else                  { /*            ...            */ }
    });
} catch (err) {
    /* ... */
}
```

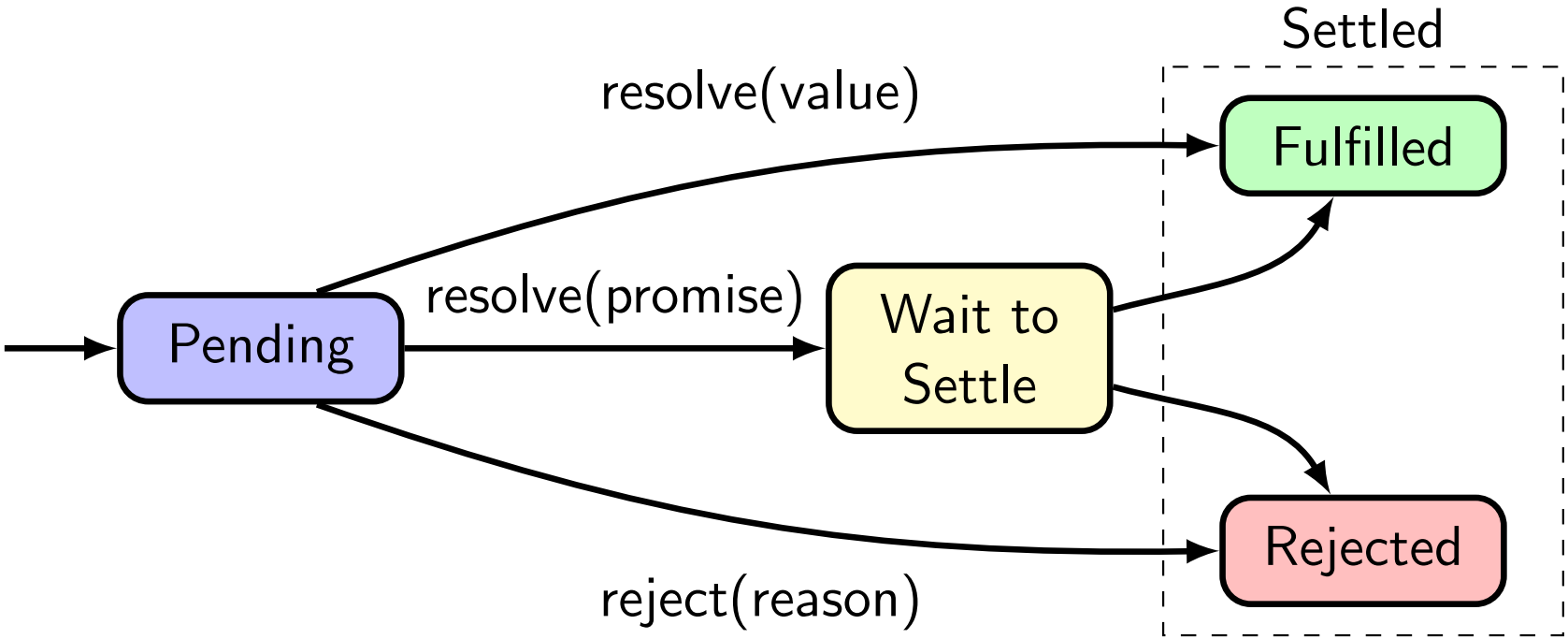
20% of function calls provide a callback

72% of callbacks are asynchronous

48% of callbacks are anonymous

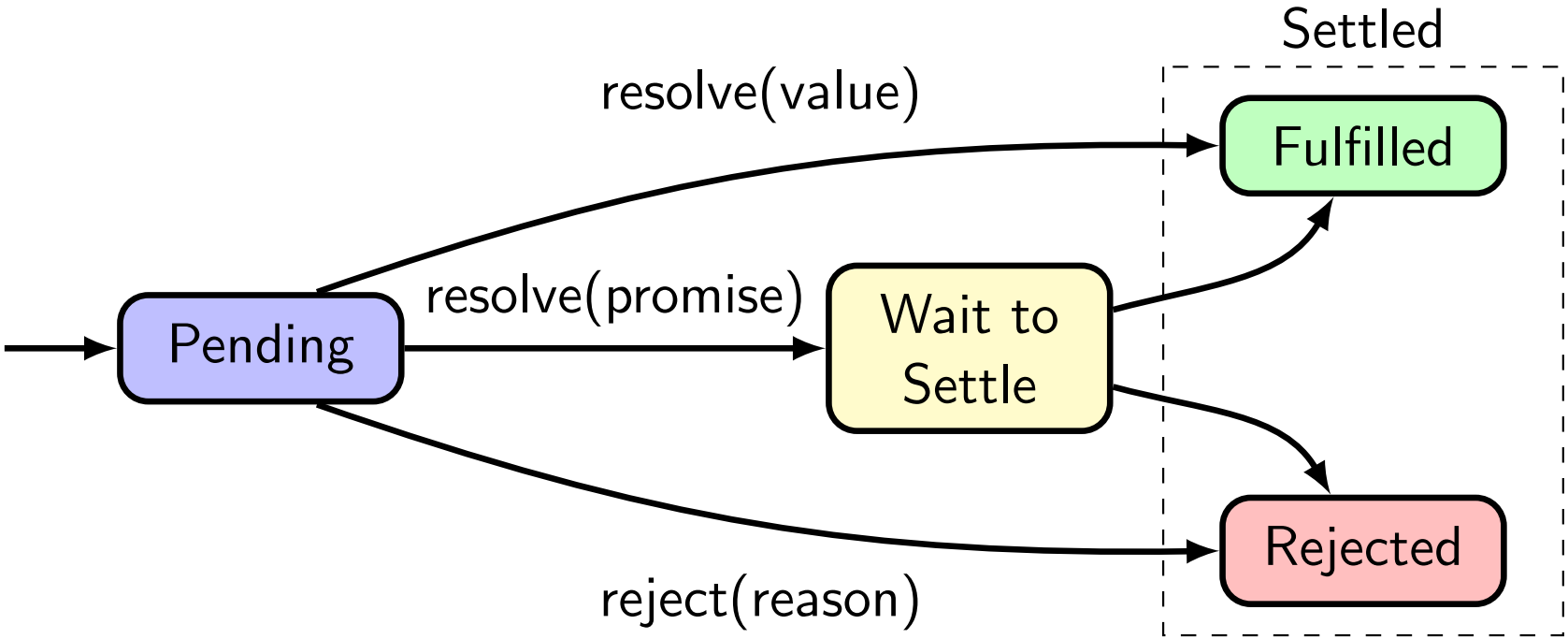
(Gallaba et al.)

Promises – A Better Abstraction




```
promise.then(onSuccess);
promise.then(onSuccess, onError);
promise.then(onSuccess).catch(onError);
```

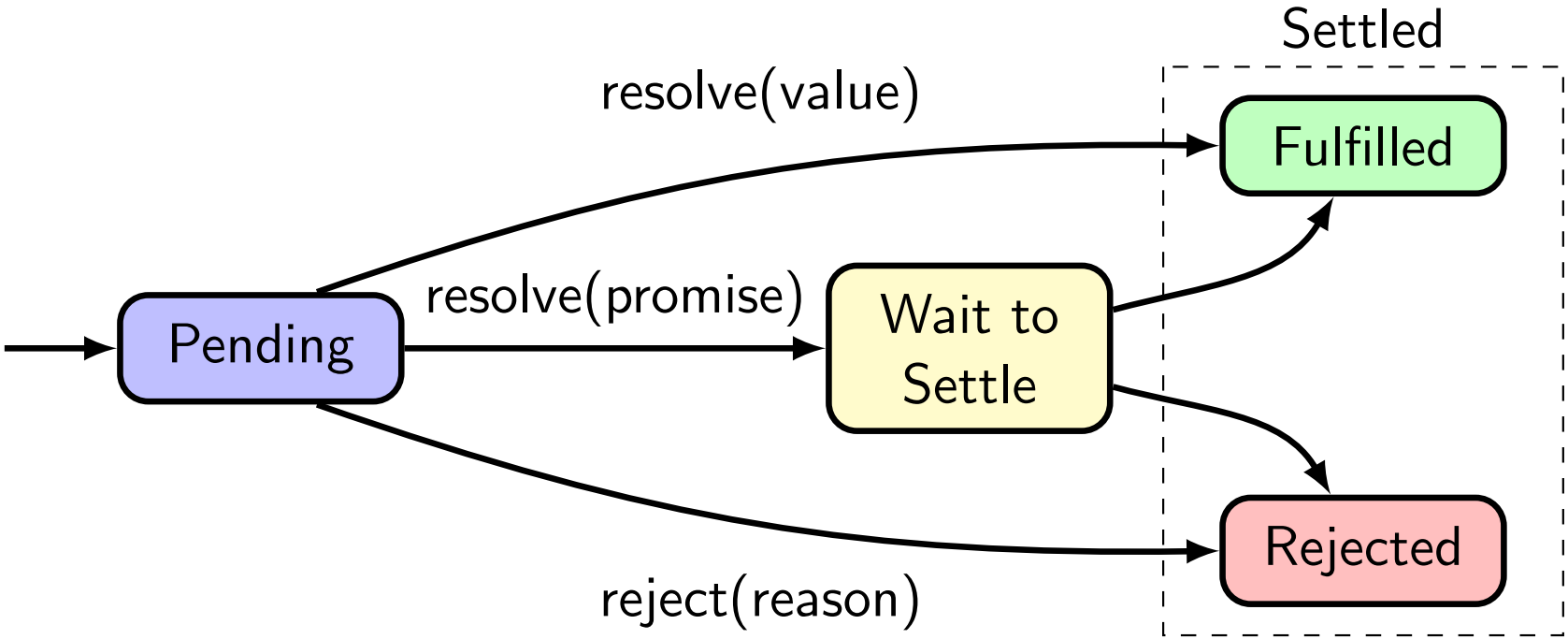
Promises – A Better Abstraction




```
var promise = new Promise((resolve, reject) => {
    conn.query('SELECT * from items', (err, row) => {
        if (!!err) resolve(row);
        else       reject(err);
    });
});
```

```
promise.then(onSuccess);
promise.then(onSuccess, onError);
promise.then(onSuccess).catch(onError);
```

Promises – A Better Abstraction

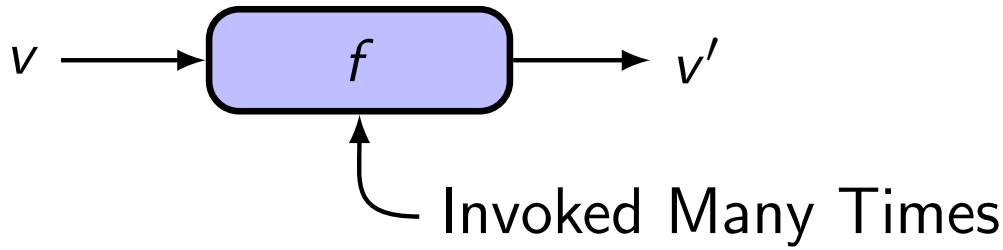



```
var promise = new Promise((resolve, reject) => {
    conn.query('SELECT * from items', (err, row) => {
        if (!!err) resolve(row);
        else       reject(err);
    });
});
```

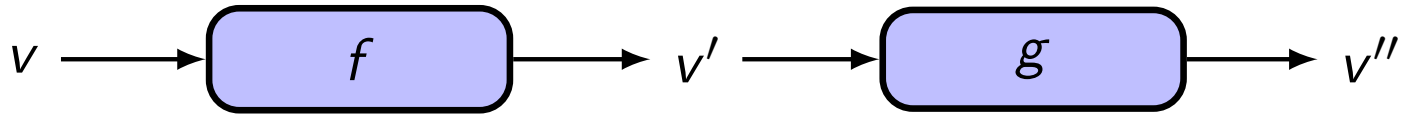
```
promise.then(onSuccess);
promise.then(onSuccess, onError);
promise.then(onSuccess).catch(onError);
```

Arrows – Attacking the Abstraction Differently

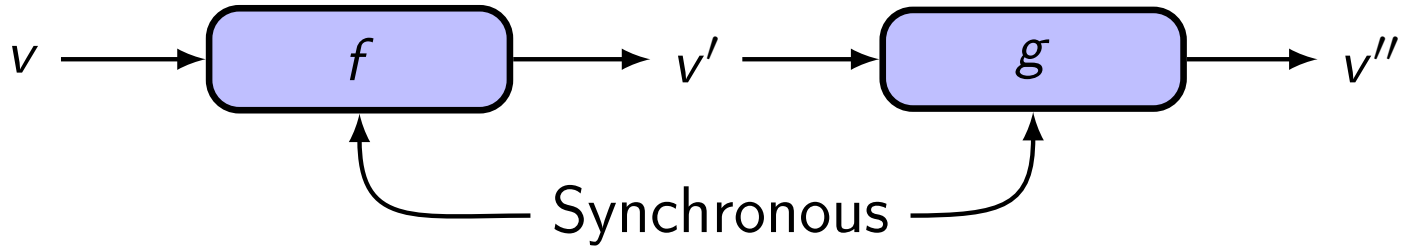
Arrows – Attacking the Abstraction Differently



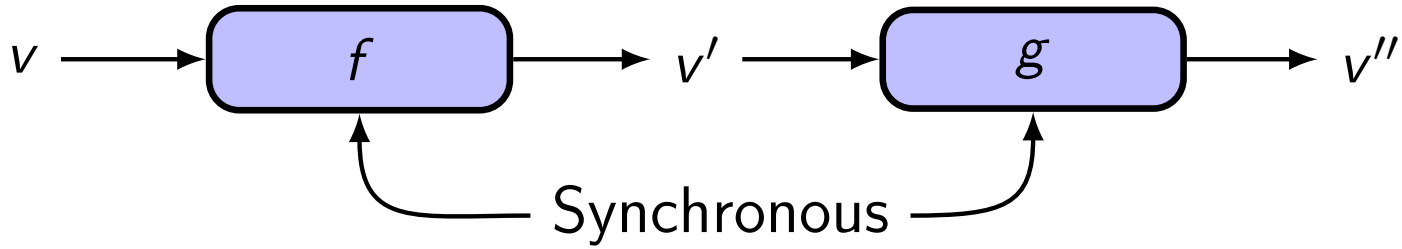
Arrows – Attacking the Abstraction Differently

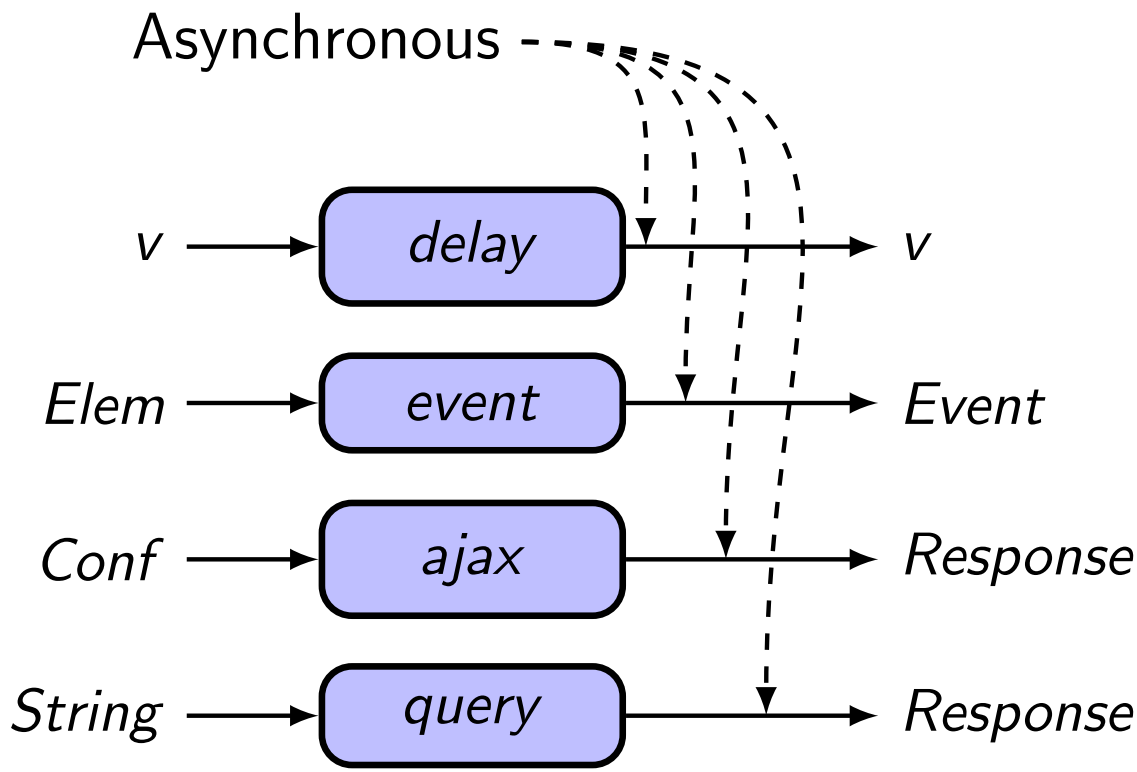


Arrows – Attacking the Abstraction Differently



Arrows – Attacking the Abstraction Differently





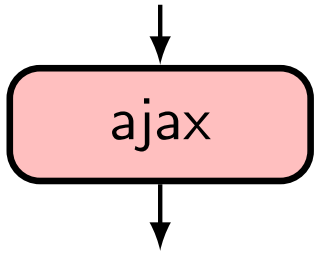
Arrows - Async Machine Composition

Composition Time

Execution Time

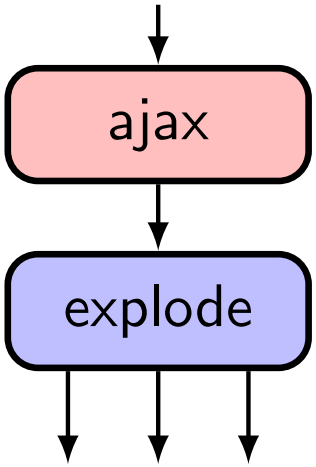
```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

Arrows - Async Machine Composition



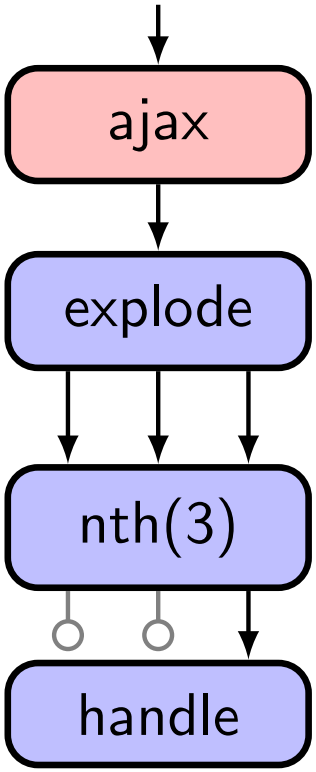

```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

Arrows - Async Machine Composition



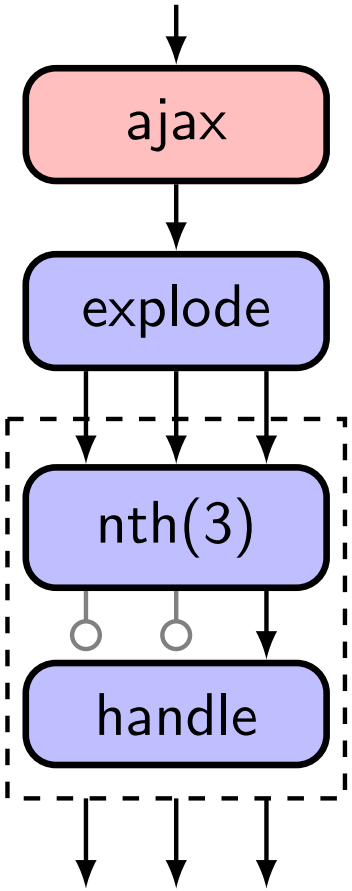

```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

Arrows - Async Machine Composition



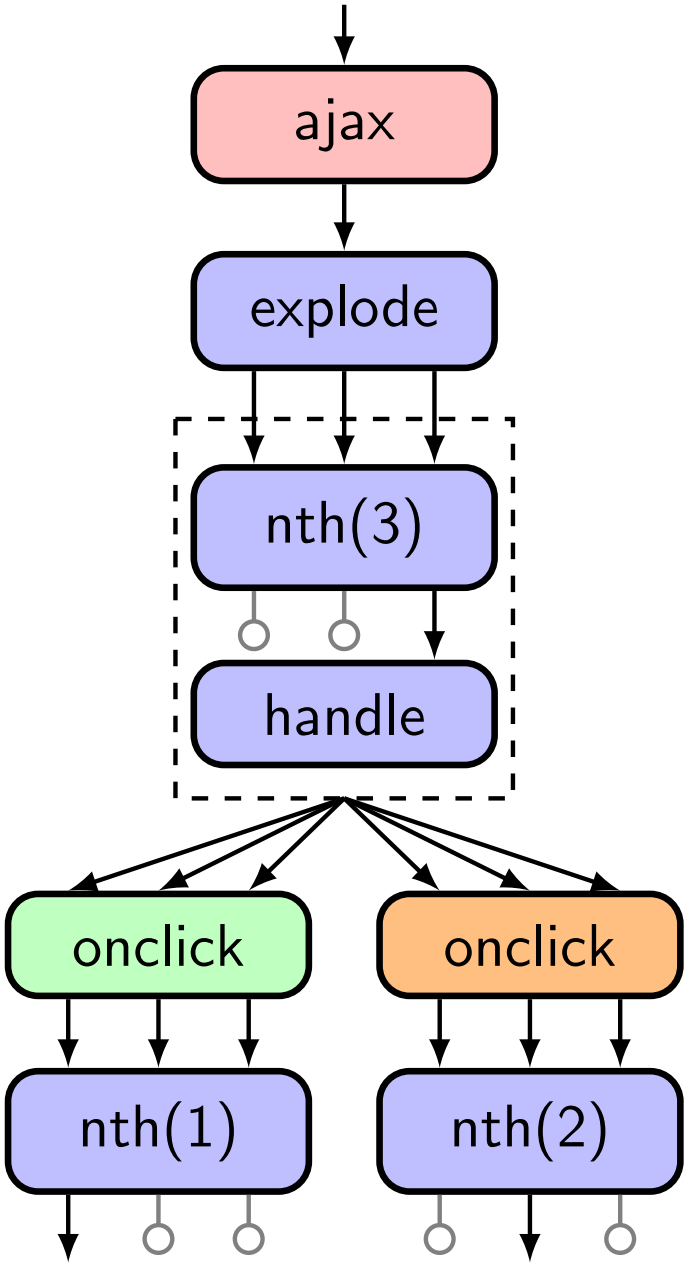

```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

Arrows - Async Machine Composition



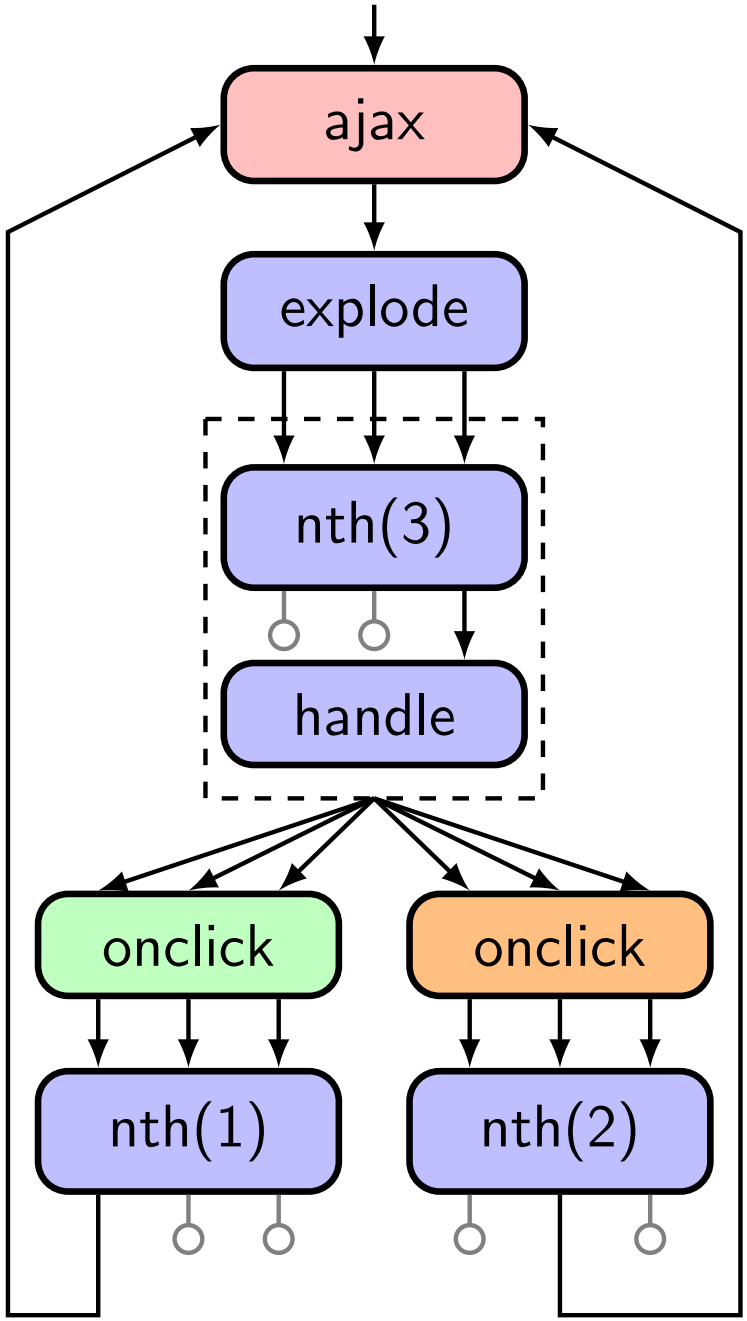

```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

Arrows - Async Machine Composition



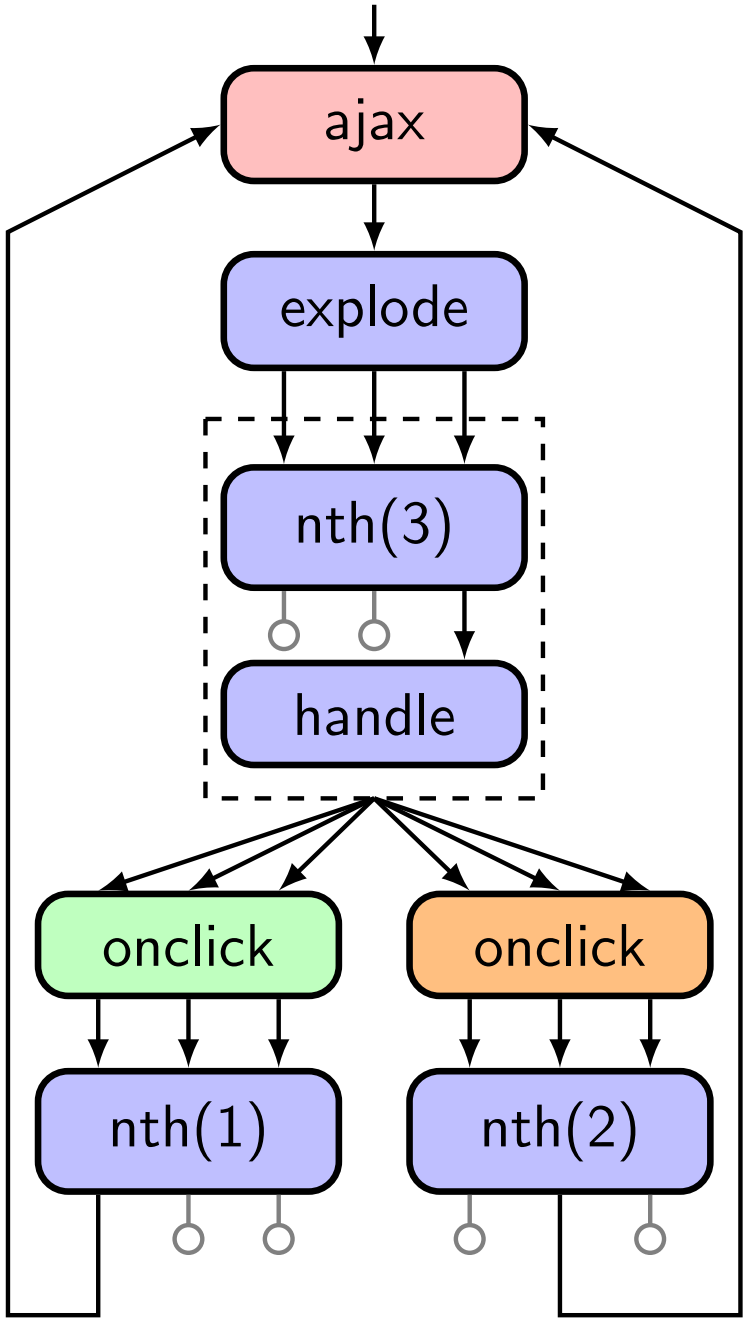

```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

Arrows - Async Machine Composition




```
let page = Arrow.fix(a => Arrow.seq([
  ajax,
  explode,
  Arrow.seq([
    new NthArrow(3),
    handle
  ]).remember(),
  Arrow.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).onclick('#next'),
  ]),
  a,
]));

let filter = Arrow.fix(a => Arrow.any([
  Arrow.seq([
    getQueryAndPage,
    page
  ]).noemit(),
  a
]).onkeyup('#filter'));

filter.run();
```

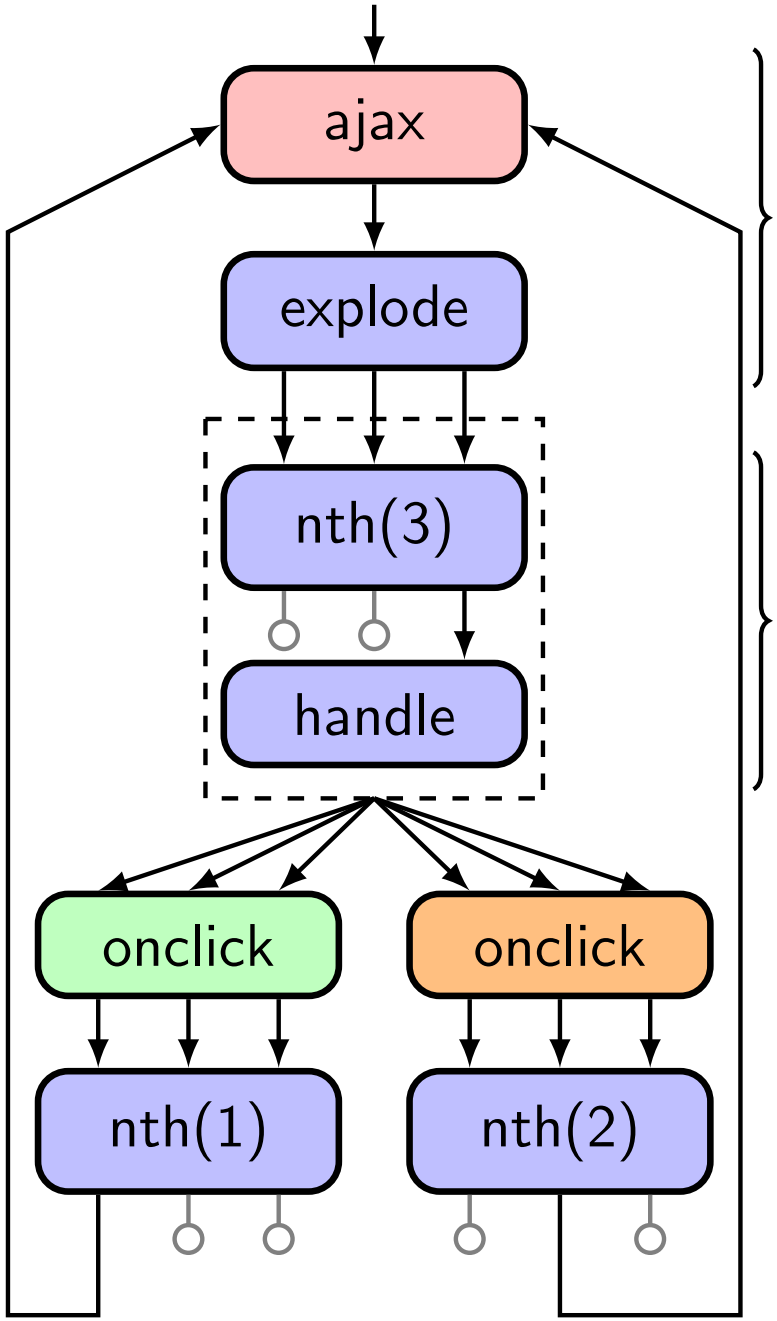
Composition-Time Inference



$(Str, Num) \rightsquigarrow$
$((Str, Num),\ (Str, Num),\ [Item])$

$[Item] \rightsquigarrow \top$

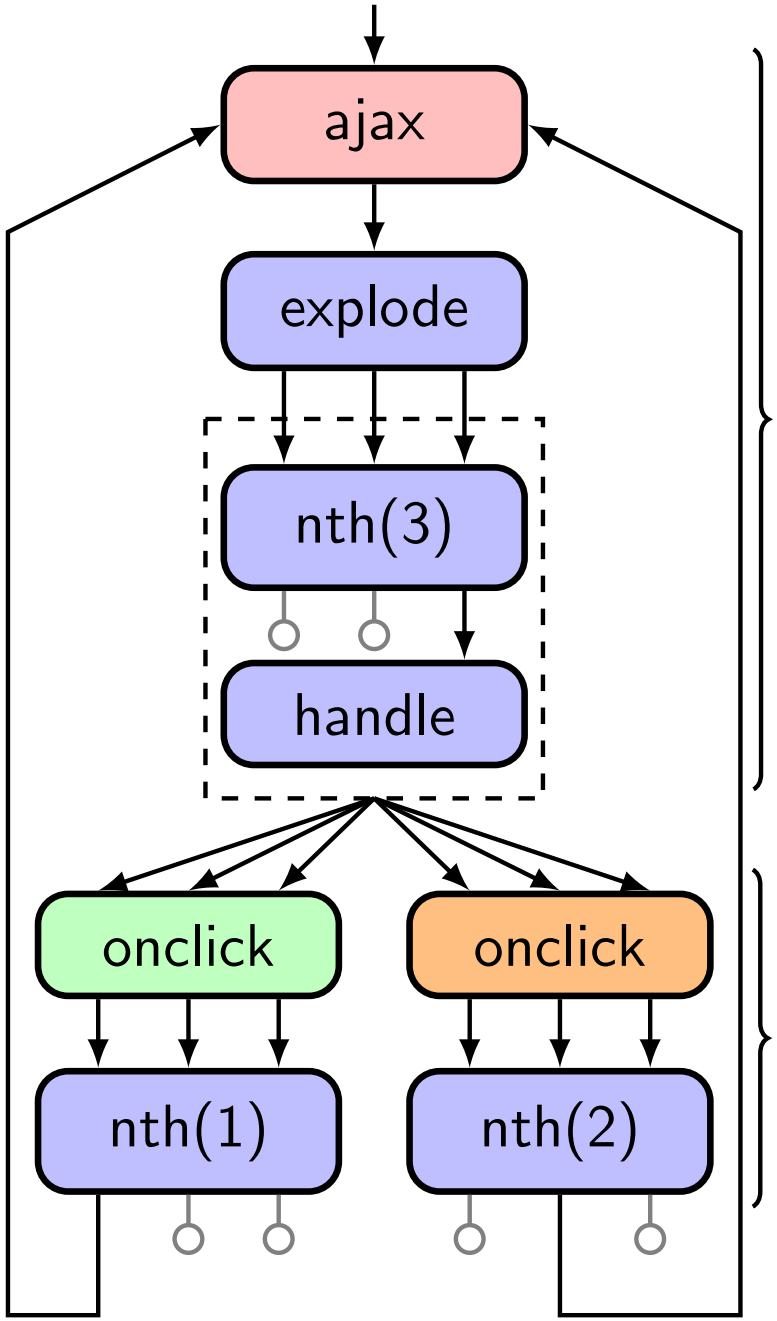
Composition-Time Inference



$(Str, Num) \rightsquigarrow$
$((Str, Num),\ (Str, Num),\ [Item])$

$(\alpha,\ \beta,\ [Item]) \rightsquigarrow (\alpha,\ \beta,\ [Item])$

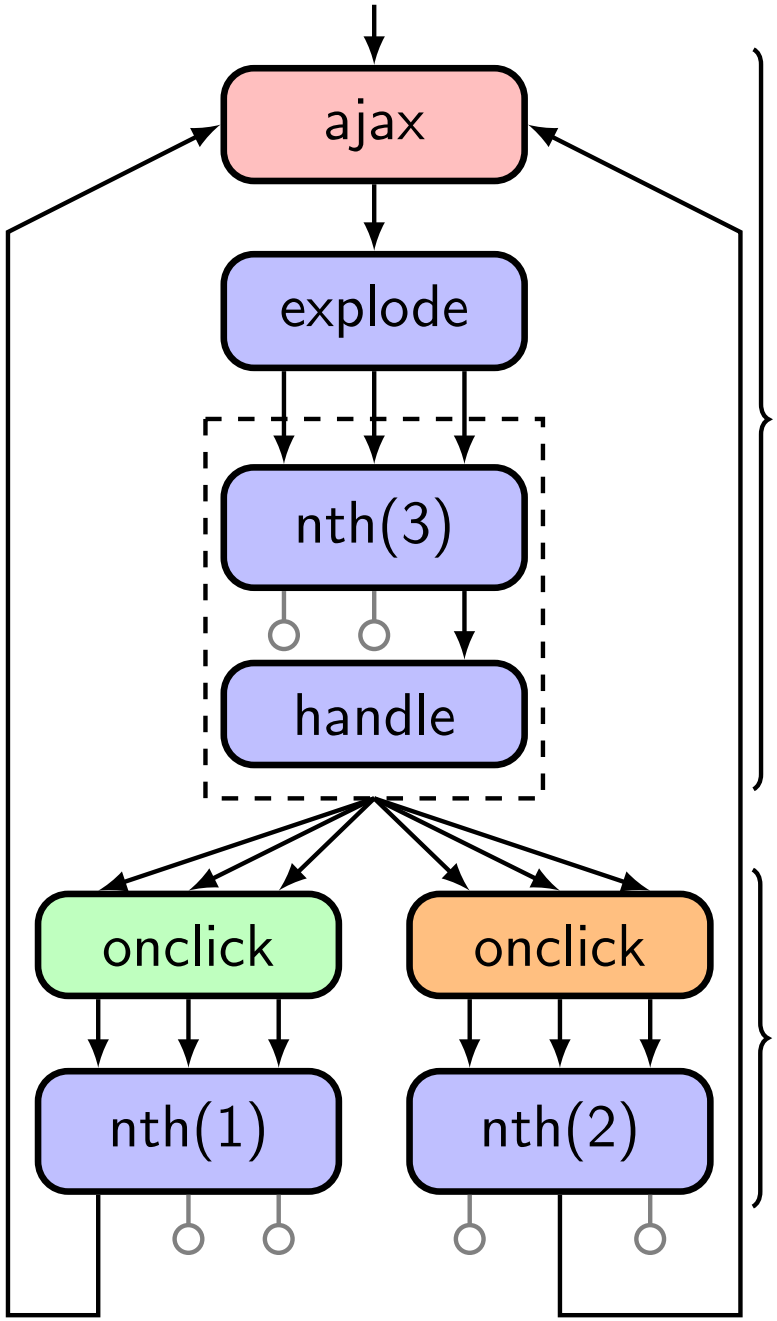
Composition-Time Inference



$(Str, Num) \rightsquigarrow$
$((Str, Num),\ (Str, Num),\ [Item])$

$(\mu,\ \nu,\ \theta) \rightsquigarrow \mu \setminus \{\mu \leq (Str, Num)\}$ and
$(\mu,\ \nu,\ \theta) \rightsquigarrow \nu \setminus \{\nu \leq (Str, Num)\}$

Composition-Time Inference

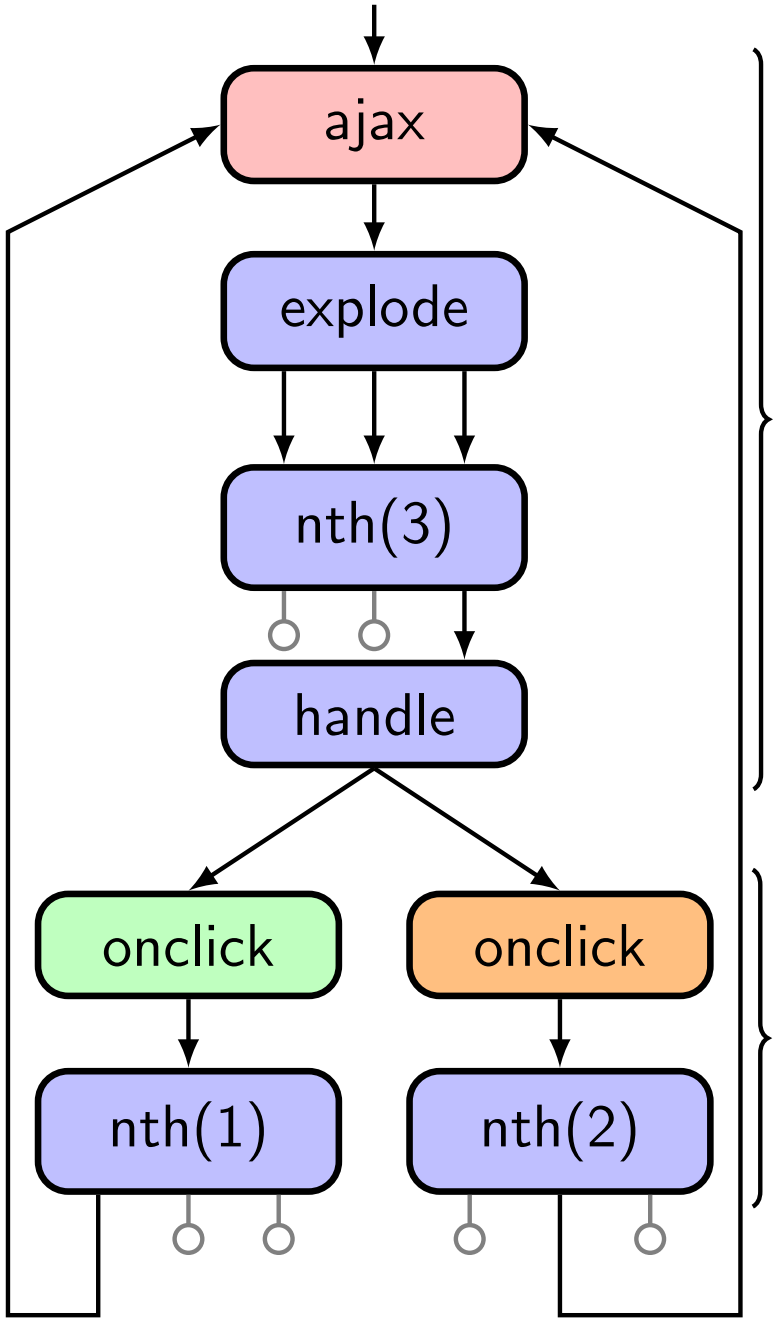


$(Str, Num) \rightsquigarrow$
$((Str, Num),\ (Str, Num),\ [Item])$

$(Str, Num) \leq \nu \leq (Str, Num)$
$(Str, Num) \leq \mu \leq (Str, Num)$

$(\mu,\ \nu,\ \theta) \rightsquigarrow \mu \setminus \{\mu \leq (Str, Num)\}$ and
$(\mu,\ \nu,\ \theta) \rightsquigarrow \nu \setminus \{\nu \leq (Str, Num)\}$

Type Clash Detection



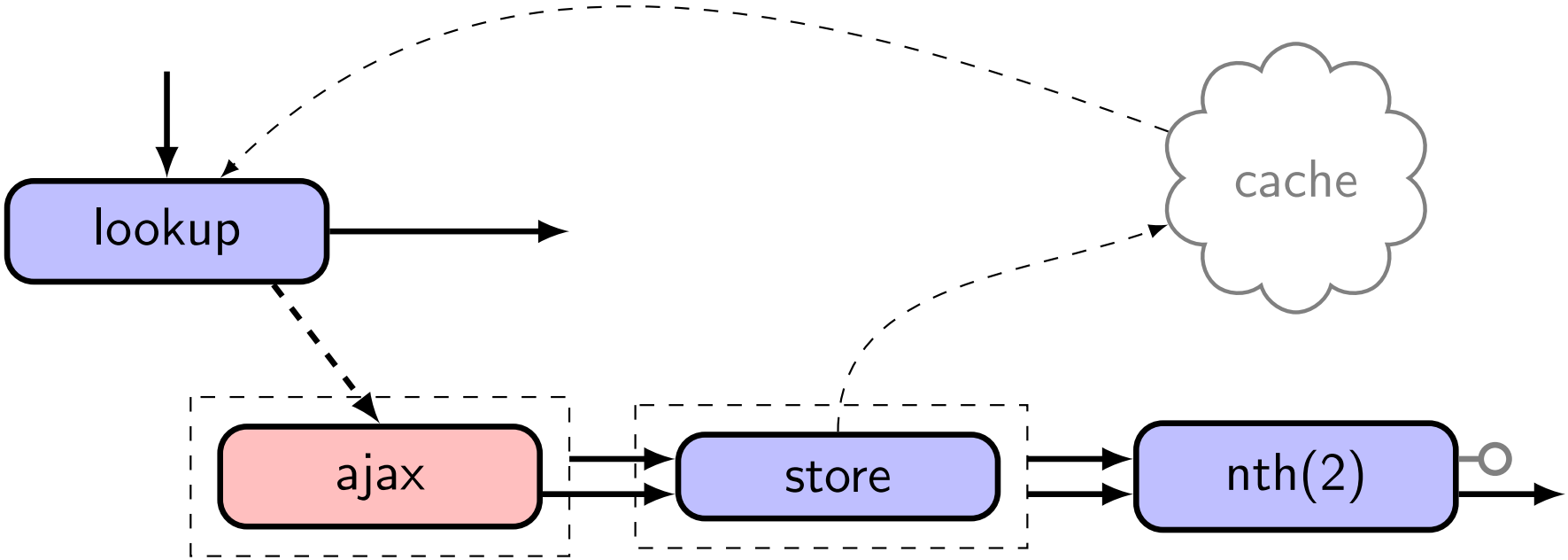
$(Str, Num) \rightsquigarrow \top$

$\top \not\leq (\mu,\ \nu,\ \theta)$

$(\mu,\ \nu,\ \theta) \rightsquigarrow \mu \setminus \{\mu \leq (Str, Num)\}$ and
$(\mu,\ \nu,\ \theta) \rightsquigarrow \nu \setminus \{\nu \leq (Str, Num)\}$

Adding Optimizations

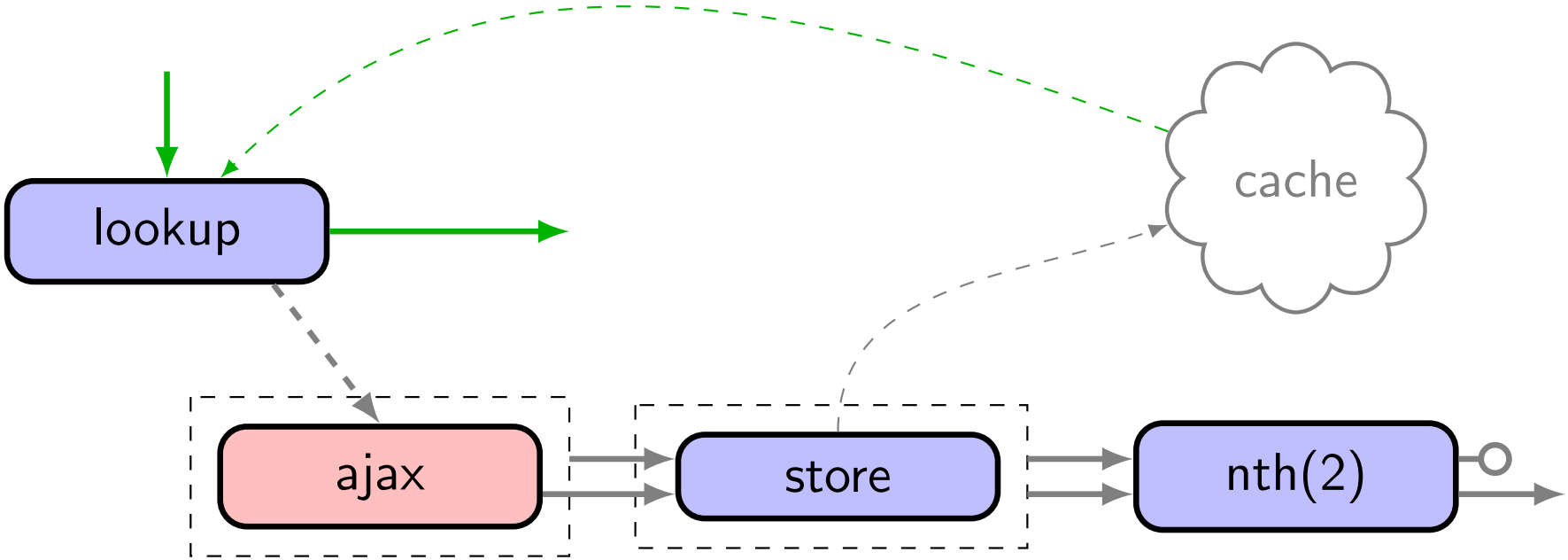
Cached Ajax Responses




```
const cachedAjax = Arrow.catch(lookup, Arrow.seq([
    ajax.carry(),
    store.remember(),
    new NthArrow(2)
]));
```

```
function lookup(key) {
    if (key in cache) return cache[key];
    else              throw key;
}
```

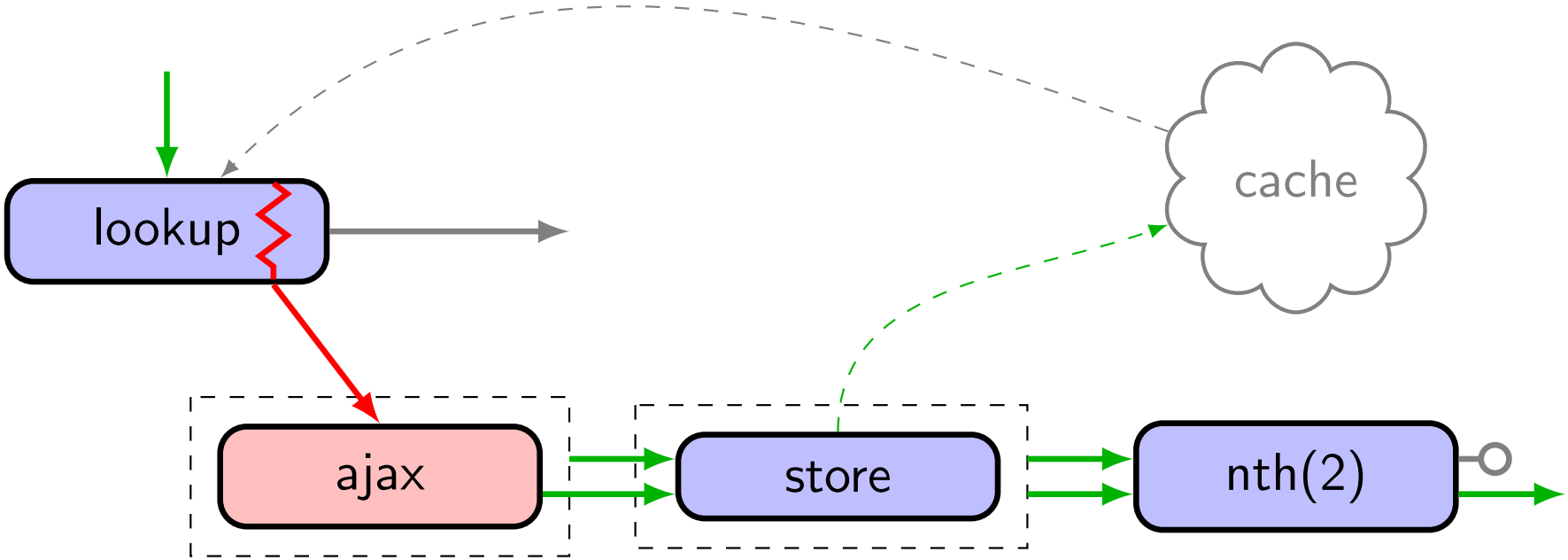
Cached Ajax Responses




```
const cachedAjax = Arrow.catch(lookup, Arrow.seq([
    ajax.carry(),
    store.remember(),
    new NthArrow(2)
]));
```

```
function lookup(key) {
    if (key in cache) return cache[key];
    else              throw key;
}
```

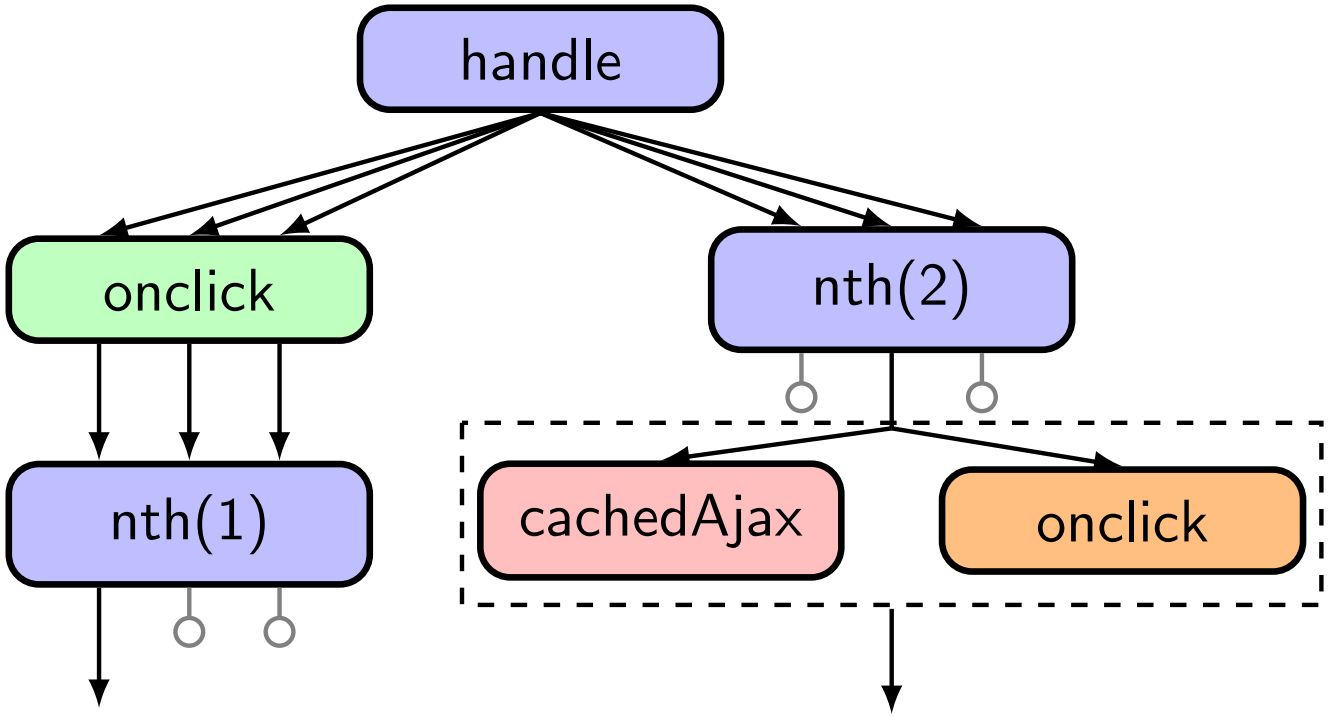
Cached Ajax Responses




```
const cachedAjax = Arrow.catch(lookup, Arrow.seq([
    ajax.carry(),
    store.remember(),
    new NthArrow(2)
]));
```

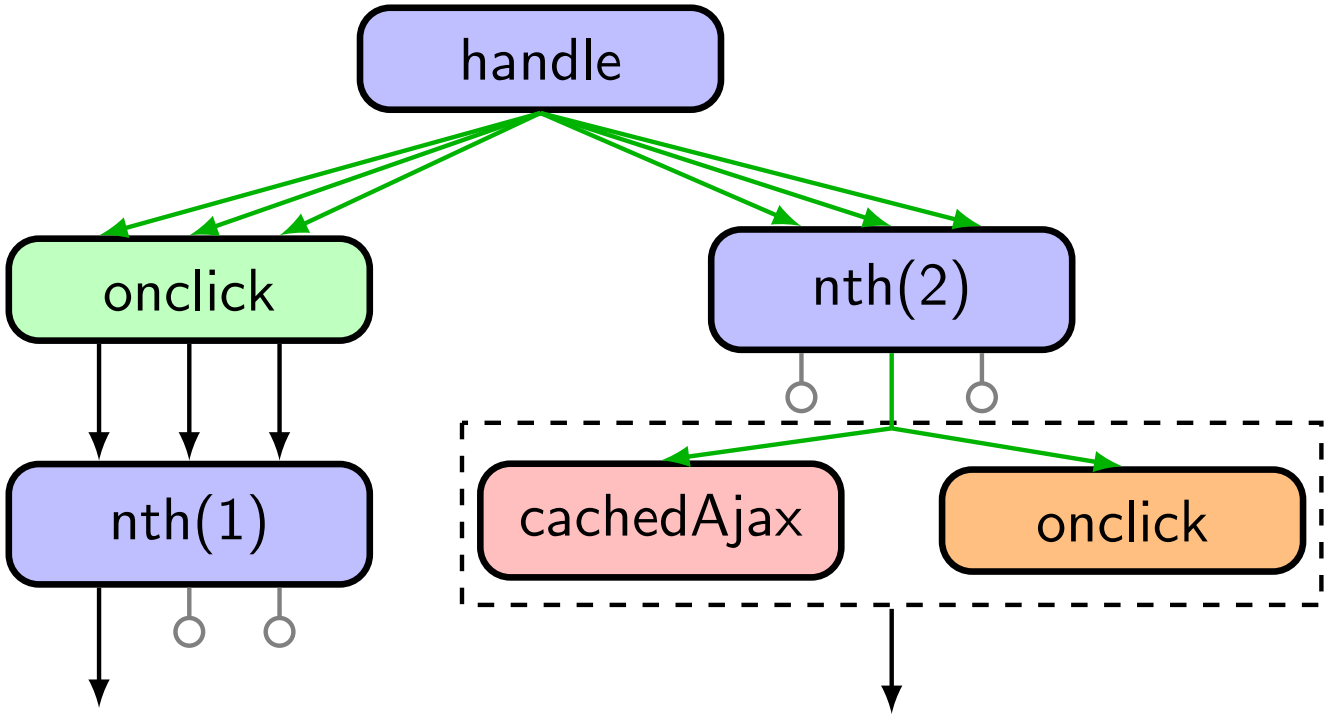
```
function lookup(key) {
    if (key in cache) return cache[key];
    else              throw key;
}
```

Pre-Fetching Next Page



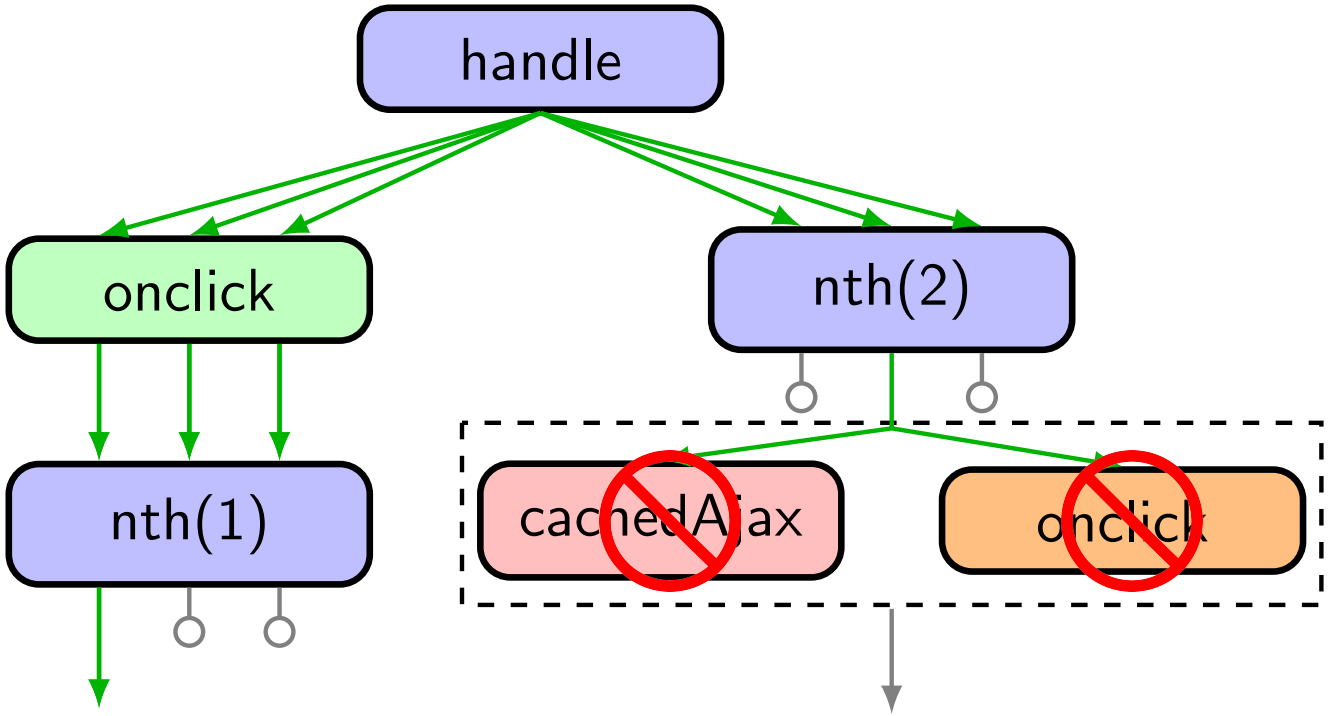

```
Arrows.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).seq(Arrow.fanout([
        ajaxOrCached,
        onclick('#next')
    ]).noemit().remember())
]);
```

Pre-Fetching Next Page



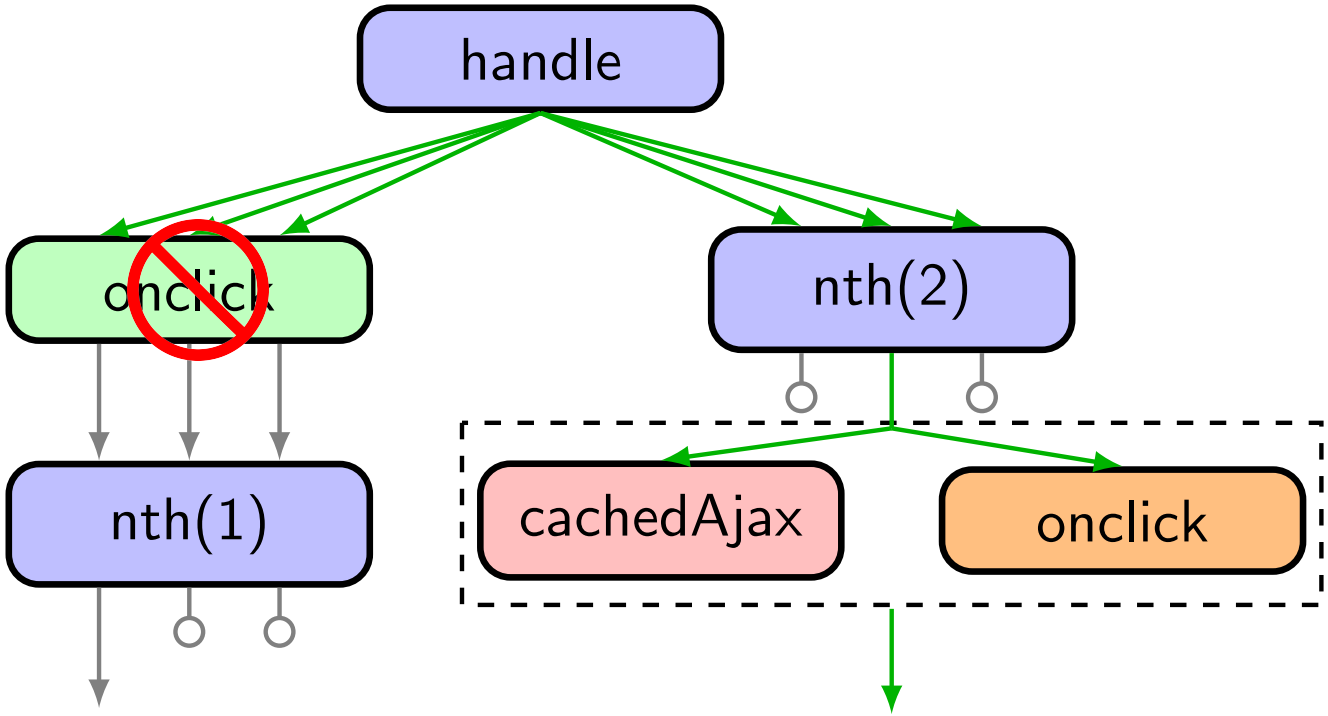

```
Arrows.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).seq(Arrow.fanout([
        ajaxOrCached,
        onclick('#next')
    ]).noemit().remember())
]);
```

Pre-Fetching Next Page



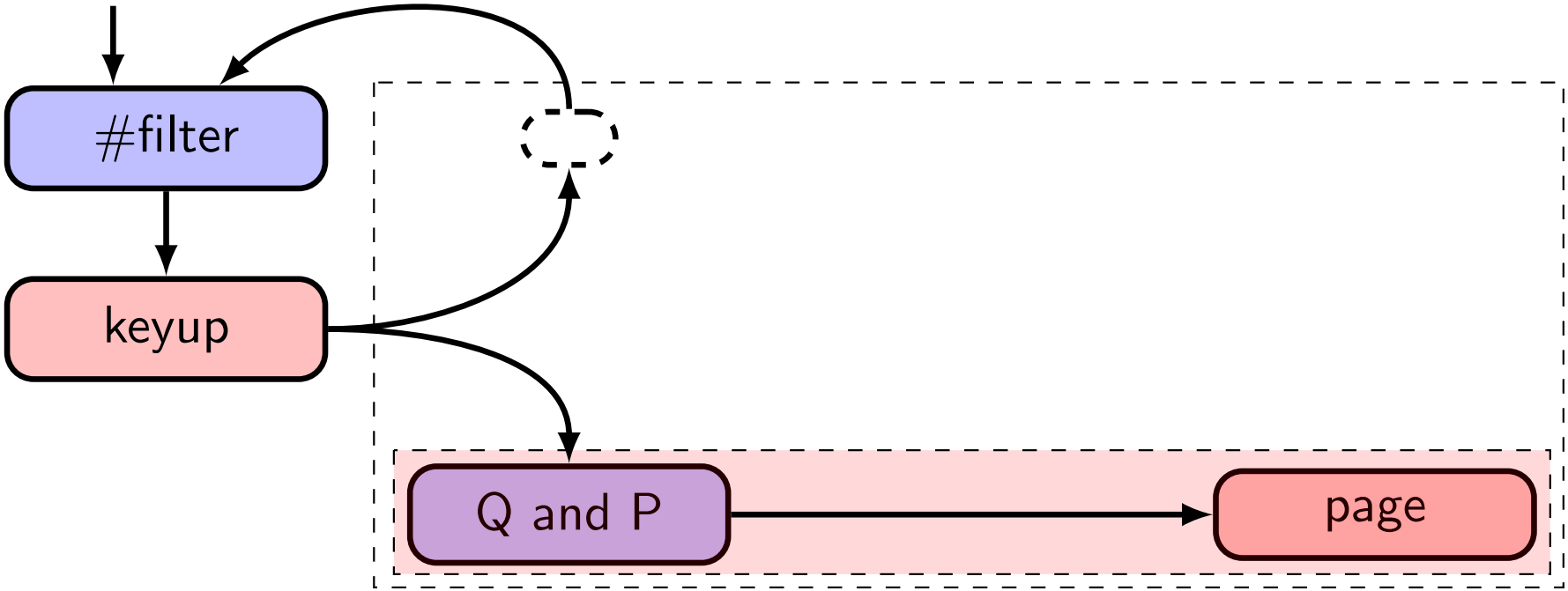

```
Arrows.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).seq(Arrow.fanout([
        ajaxOrCached,
        onclick('#next')
    ]).noemit().remember())
]);
```

Pre-Fetching Next Page



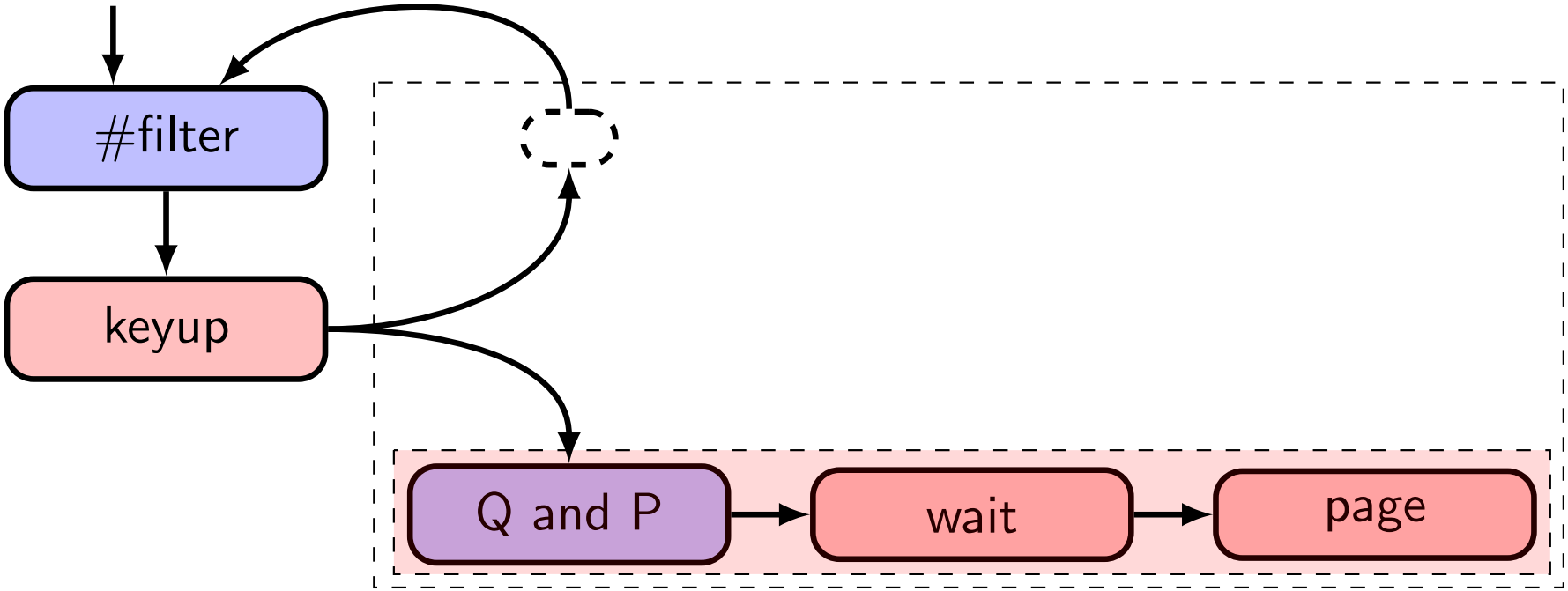

```
Arrows.any([
    new NthArrow(1).onclick('#prev'),
    new NthArrow(2).seq(Arrow.fanout([
        ajaxOrCached,
        onclick('#next')
    ]).noemit().remember())
]);
```

No In-Flight Requests as User Types



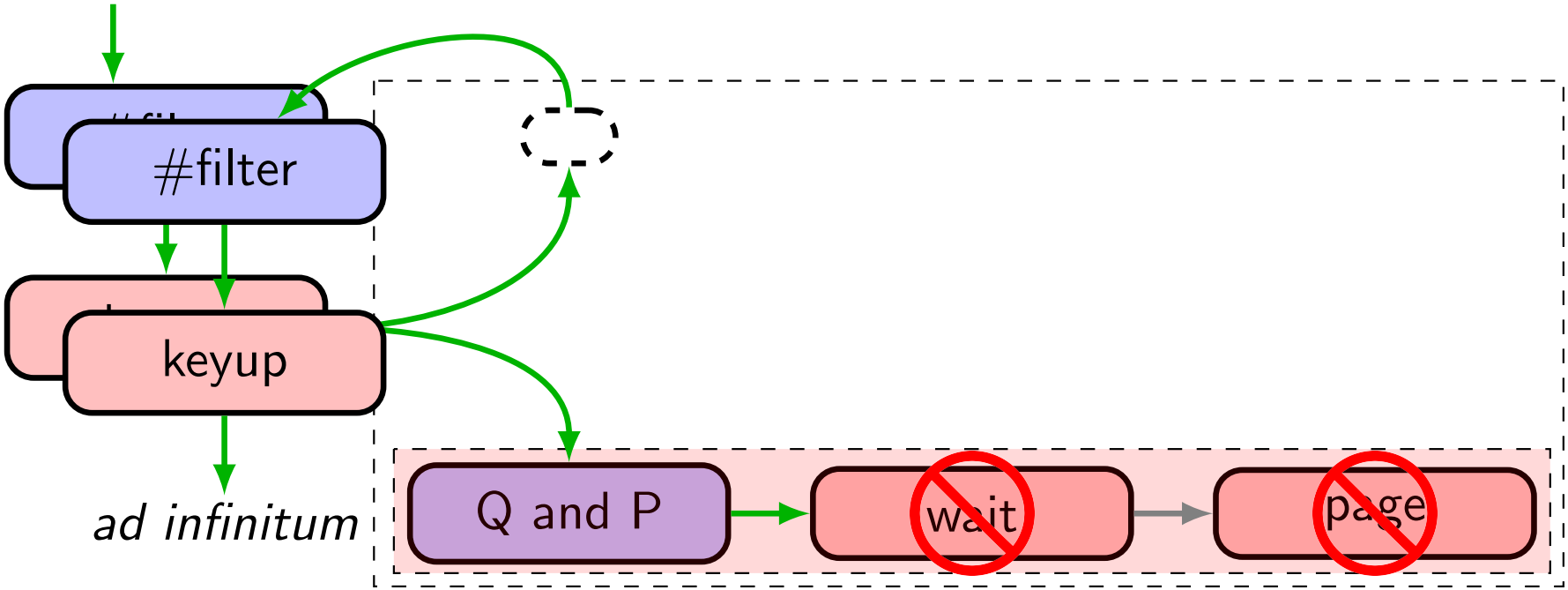

```
let filter = Arrow.fix(a => Arrow.any([
  a,
  Arrow.seq([
    getQueryAndPage,

    page
  ]).noemit(),
]).onkeyup('#filter'));
```

No In-Flight Requests as User Types




```
let filter = Arrow.fix(a => Arrow.any([
  a,
  Arrow.seq([
    getQueryAndPage,
    new DelayArrow(400),
    page
  ]).noemit(),
]).onkeyup('#filter'));
```

No In-Flight Requests as User Types




```
let filter = Arrow.fix(a => Arrow.any([
  a,
  Arrow.seq([
    getQueryAndPage,
    new DelayArrow(400),
    page
  ]).noemit(),
]).onkeyup('#filter'));
```

Conclusion

Composing an *interaction machine* is a powerful shift in abstraction
Opens up opportunity for ‘static’ benefits

Conclusion

Composing an *interaction machine* is a powerful shift in abstraction
Opens up opportunity for 'static' benefits

Examples, papers, and code available at
`http://arrows.eric-fritz.com`

Conclusion

Composing an *interaction machine* is a powerful shift in abstraction
Opens up opportunity for 'static' benefits

Examples, papers, and code available at
`http://arrows.eric-fritz.com`

Questions?