ENTERPRISE SCHEDULING WITH HASKELL

Ryan Trinkle, CTO
skedge.me
SKEDGE.ME

- Cloud-based scheduling platform
- Branded customer experience
- Flexible business workflow
- Deep integration
  - Backend-systems
  - In-store workflow (iPad, etc.)
- Clients
  - Sephora
  - Tiffany
  - Stanford
  - FBI
  - ...

Saturday, November 30, 13
MAKE A RESERVATION

LOCATION:
SEPHORA UNION SQUARE
45 E. 17th Street
New York
New York
10003

SELECTED SERVICE:
CUSTOM MAKEOVER
Personalize your complete, full-face makeover in this dedicated session with an expert artist.
More Info

CHANGE SERVICE:
CUSTOM MAKEOVER - 45m
PERSONAL BEAUTY CONSULTATION - 1h 30m

FIND AN AVAILABLE APPOINTMENT

SUN SEP 22
MORNING
12:30 PM
1:15 PM
1:30 PM
1:45 PM
2:00 PM
2:15 PM
3:00 PM
EVENING

MON SEP 23
MORNING
AFTERNOON
EVENING

TUE SEP 24
MORNING
AFTERNOON
EVENING

NEXT STEP
MAKE A RESERVATION

LOCATION:
SEPHORA UNION SQUARE
45 E. 17th Street
New York
New York
10003

SELECTED SERVICE:
CUSTOM MAKEOVER
Personalize your complete, full-face makeover in this dedicated session with an expert artist.
More Info

CHANGE SERVICE:
- CUSTOM MAKEOVER - 45m
- PERSONAL BEAUTY CONSULTATION - 1h 30m

FIND AN AVAILABLE APPOINTMENT

09/22/2013

SUN SEP 22
MORNING
12:30 PM
1:30 PM
2:00 PM
2:30 PM

AFTERNOON
1:15 PM
1:45 PM
2:15 PM

EVENING
3:00 PM

MON SEP 23
MORNING

AFTERNOON

EVENING

TUE SEP 24
MORNING

AFTERNOON

EVENING
PROBLEM

- 43,000 lines of Groovy on Grails
- Major, intractable bugs
  - Timezones
  - Recurring events
  - Notifications
  - Double-booking
- Poorly documented
  - Bad performance
  - Inflexible
# ARCHITECTURE

<table>
<thead>
<tr>
<th>General-purpose</th>
<th>Client-specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appointments</td>
<td>BuyNLarge</td>
</tr>
<tr>
<td>People</td>
<td>AcmeU</td>
</tr>
<tr>
<td>Notifications</td>
<td>...</td>
</tr>
</tbody>
</table>

| Biz             | Business rules enforcement |
| Sec             | Authorization             |
| DB              | ADTs, CRUD, caching, validation |
| RawDB           | ACID transactions, no side effects, automatic retries |
| IO              | Basic Haskell - unsafe side effects; anything goes |
## Architecture

<table>
<thead>
<tr>
<th></th>
<th>General-purpose</th>
<th></th>
<th>Client-specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appointments</td>
<td>Biz</td>
<td>Sec</td>
<td>BuyNLarge</td>
</tr>
<tr>
<td></td>
<td>Sec</td>
<td>DB</td>
<td>AcmeU</td>
</tr>
<tr>
<td></td>
<td>RawDB</td>
<td>IO</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
SECURITY

• Roles by client
  • Basic: Owner, Staff, Customer
  • BuyNLarge: Leadership, District Manager, Store Manager, Sales Associate, Customer
  • AcmeU: Admin, Faculty, Staff, Teaching Assistant, Student
SECURITY

• Verbs by component

  • Appointment: Book, Join, Reschedule, Cancel, Delete
  • People: Create, Read, Update, Delete
  • Notifications: Send, Edit, Cancel, etc.
  • BuyNLarge Reports: Run District Report, Run Store Report
class Policy role verb where

    isAuthorized :: role -> verb -> DB Bool
instance Policy BNLRole BNLReportVerb where

    isAuthorized BNLLeadership _ = return True

    isAuthorized (BNLDistrictManager dist) a = case a of
        BNLRunDistrictReport dist’ -> return (dist == dist’)
        BNLRunStoreReport store -> return (storeDistrict store == dist)

    isAuthorized (BNLStoreManager store) a = case a of
        BNLRunStoreReport store’ -> return (store == store’)
        _ -> return False

    isAuthorized _ _ = return False
## SECURITY

<table>
<thead>
<tr>
<th>General-purpose</th>
<th>Client-specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appointments</td>
<td>BuyNLarge</td>
</tr>
<tr>
<td>People</td>
<td>AcmeU</td>
</tr>
<tr>
<td>Notifications</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Standard</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>BuyNLarge</th>
<th>AcmeU</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BuyNLarge</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>AcmeU</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>...</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
instance Policy BNLRole AppointmentVerb where

  isAuthorized r = isAuthorized $ case r of

  BNLLeadership -> StandardAdmin
  BNLDistrictManager _ -> StandardStaff
  BNLStoreManager _ -> StandardStaff
  BNLCustomer -> StandardCustomer
# SECURITY

<table>
<thead>
<tr>
<th>General-purpose</th>
<th>Client-specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appointments</td>
<td>BuyNLarge</td>
</tr>
<tr>
<td>People</td>
<td>AcmeU</td>
</tr>
<tr>
<td>Notifications</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

| Standard        |               |               |
| BuyNLarge       |               |               |
| AcmeU           |               |               |
| ...             |               |               |

Saturday, November 30, 13
QUICK-AND-DIRTY IMPORTER

- Limited set of inputs
- “End-user” is our developers
- Thrown away when job is done
QUICK-AND-DIRTY IMPORTER

• 1400 lines of Haskell
• Never refactored
• Lots of partiality
• Lots weirdly-specific, huge functions
• Leans heavily on type system
importService :: EventNotificationPolicy -> (Maybe Bool -> Map (Ref ThingField) FieldProperties) -> (Maybe Bool -> Map (Ref AttendeeField) FieldProperties) -> Ref ThingField -> Ref ThingField -> Ref ThingField -> Ref AttendeeField -> Ref AttendeeField -> Ref ServiceField -> Ref ServiceField -> Ref Role -> Ref Role -> Ref Role -> Ref (Constraint (Ref Thing)) -> (Text -> Ref Thing) -> (Text -> Ref Thing) -> (Text -> Maybe (Ref AttendeeField, CustomAttendeeField)) -> S1.Service -> DB (Ref Sk.Service)
LIBRARIES

• Javascript: 9 libraries
• Haskell: 71 libraries + 87 dependencies
LIBRARIES

- Haskell libraries tend to be
  - Easier to find
  - Easier to vet
  - Higher quality
ANNOYANCES

• Cyclic module dependencies with hs-boot files are unwieldy - avoid
• Scalability above 4 cores is limited - apparently fixed in GHC 7.8
• Idle GC can cause issues - disable
• Debugging production issues without stack traces is hard - consider using profiling
RESULT

• 8,200 lines of Haskell
• No persistent bugs
• Greatly improved performance and flexibility
• Used daily by thousands of people