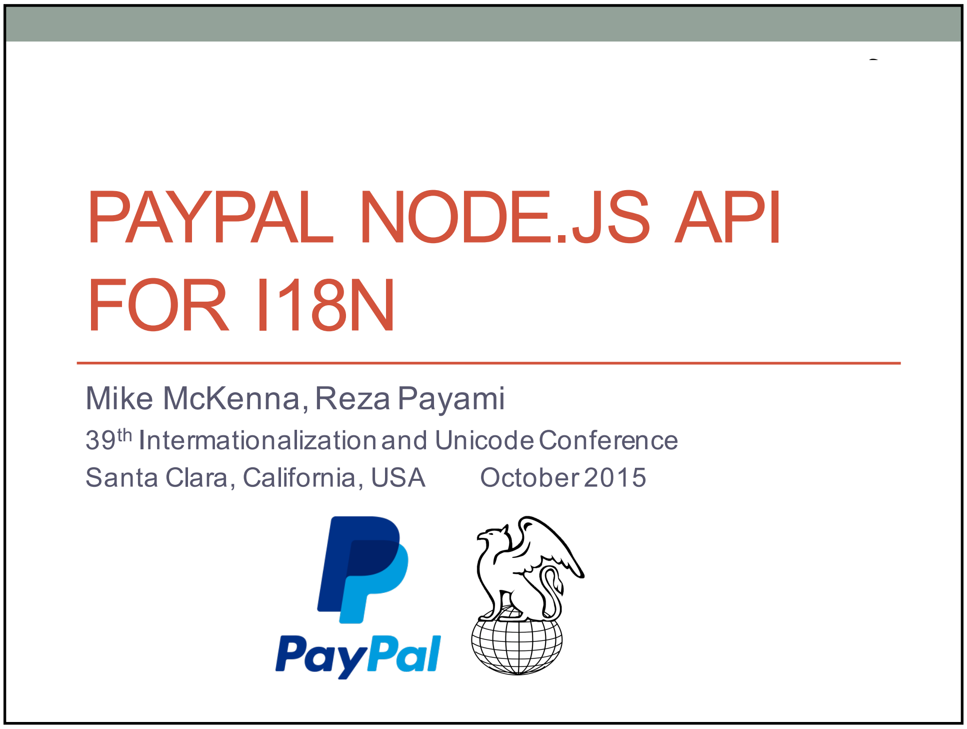
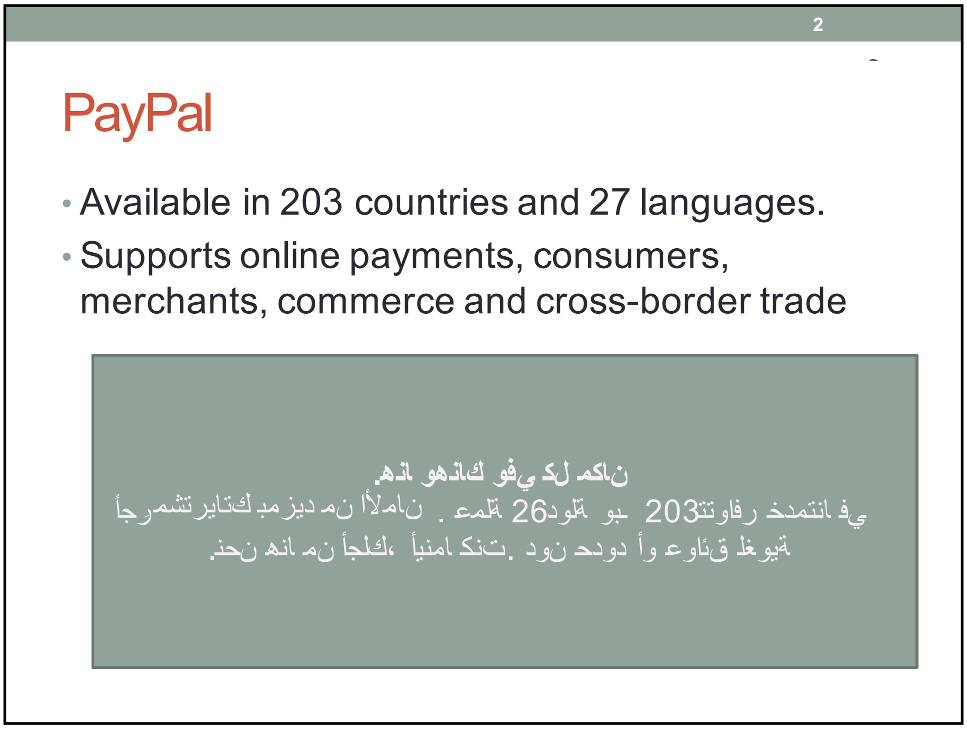


PAYPAL NODE.JS API FOR I18N

Mike McKenna, Reza Payami

39th Internationalization and Unicode Conference

Santa Clara, California, USA October 2015



2

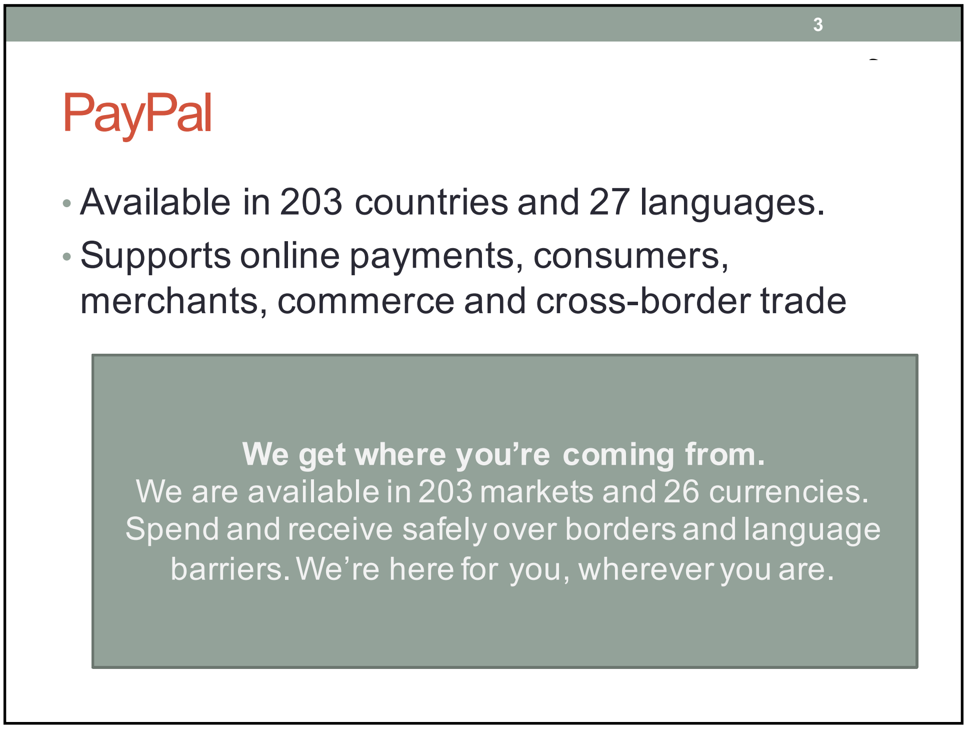
PayPal

- Available in 203 countries and 27 languages.
- Supports online payments, consumers, merchants, commerce and cross-border trade

ناكم لك يفو كانهو انه.

يف انتمدخ رفاوتت203 بو ةلود26 ةلمع . نامألا نم ديزمب كتايرتشم رِجأ

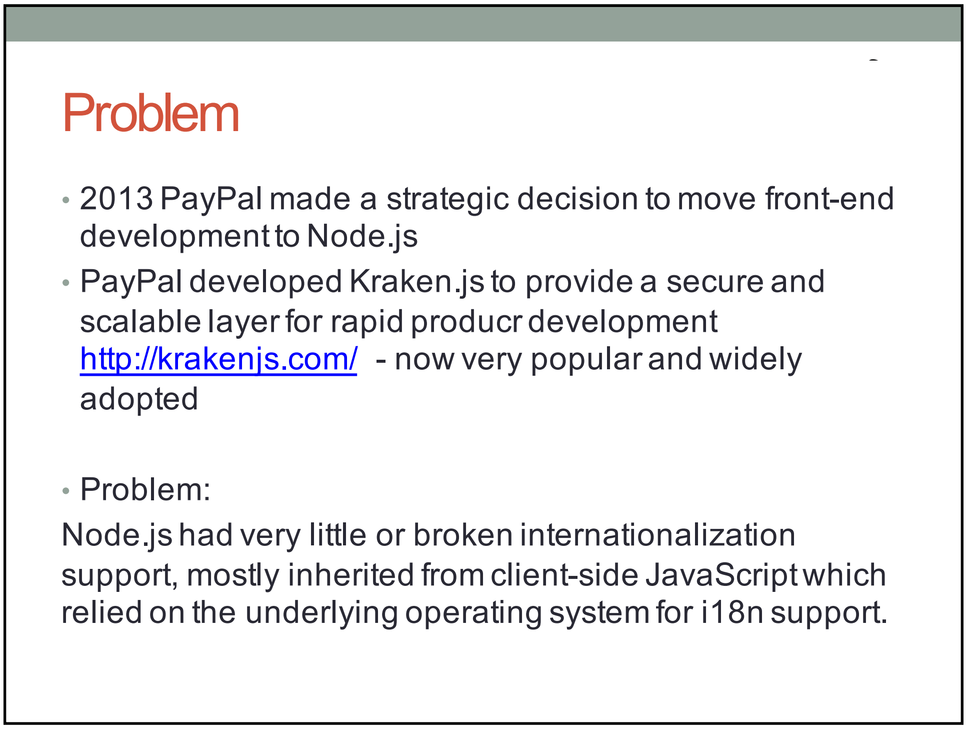
ةيوغل قئاوع وأ دودح نود .تنك امنيأ ،كلجأ نم انه نحن.



PayPal

- Available in 203 countries and 27 languages.
- Supports online payments, consumers, merchants, commerce and cross-border trade

We get where you're coming from.
We are available in 203 markets and 26 currencies. Spend and receive safely over borders and language barriers. We're here for you, wherever you are.

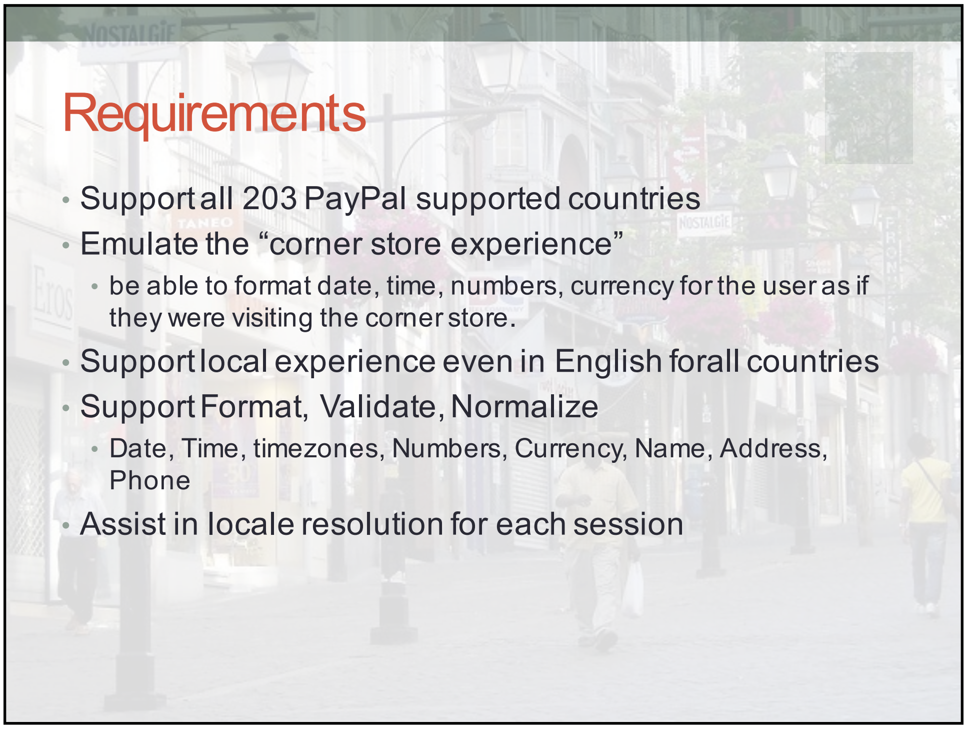


Problem

- 2013 PayPal made a strategic decision to move front-end development to Node.js
- PayPal developed Kraken.js to provide a secure and scalable layer for rapid producr development http://krakenjs.com/ - now very popular and widely adopted

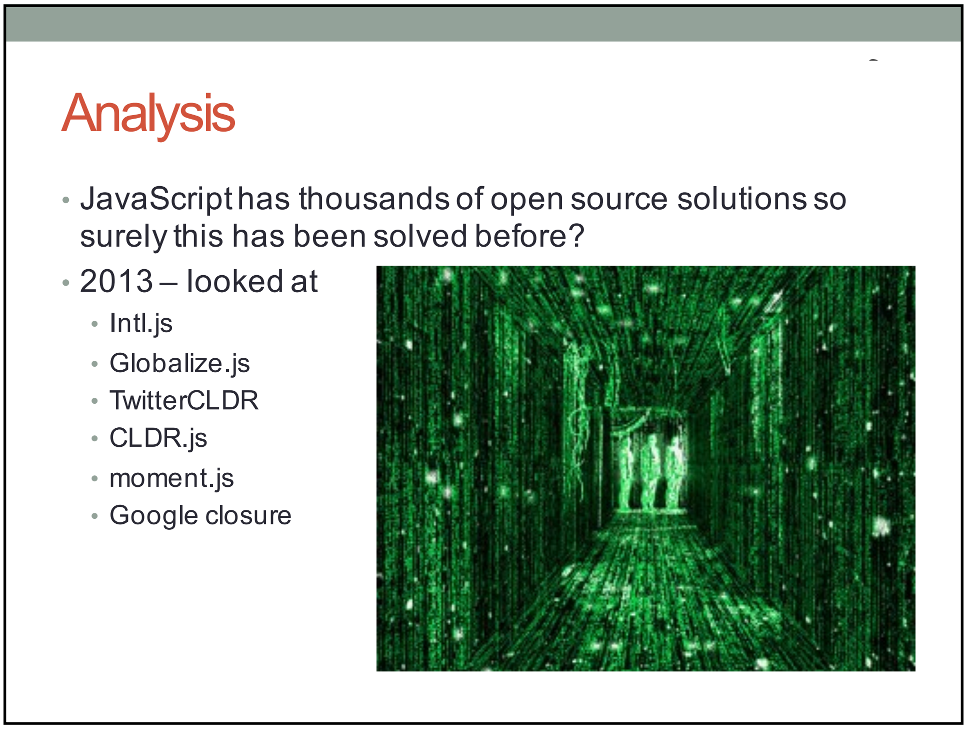
- Problem:

Node.js had very little or broken internationalization support, mostly inherited from client-side JavaScript which relied on the underlying operating system for i18n support.



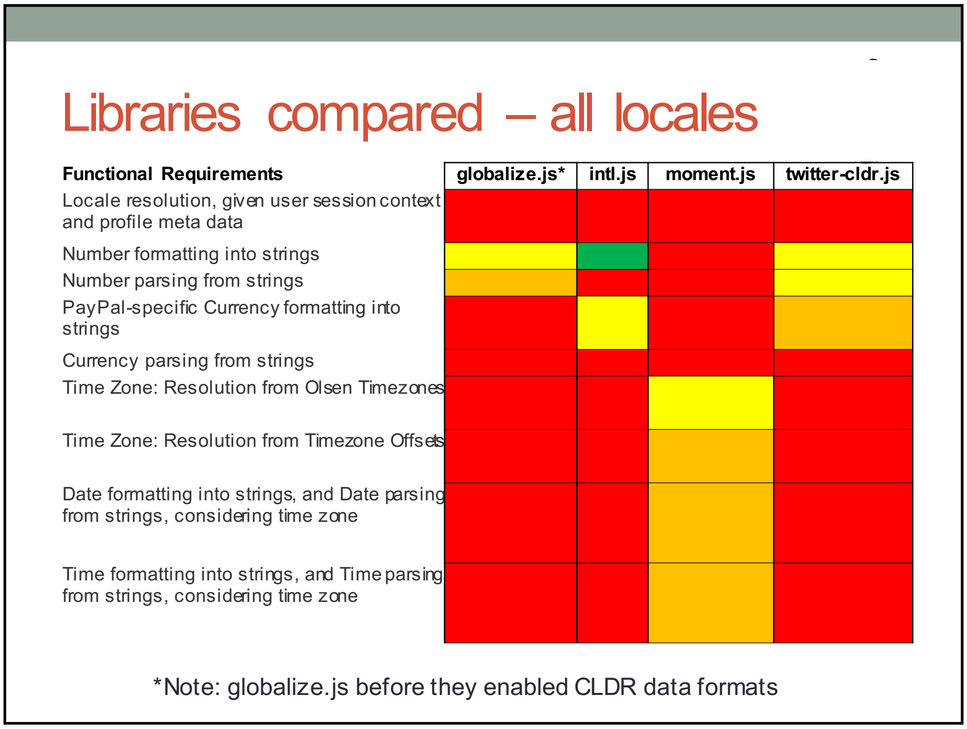
Requirements

- Support all 203 PayPal supported countries
- Emulate the “corner store experience”
  - be able to format date, time, numbers, currency for the user as if they were visiting the corner store.
- Support local experience even in English for all countries
- Support Format, Validate, Normalize
  - Date, Time, timezones, Numbers, Currency, Name, Address, Phone
- Assist in locale resolution for each session



Analysis

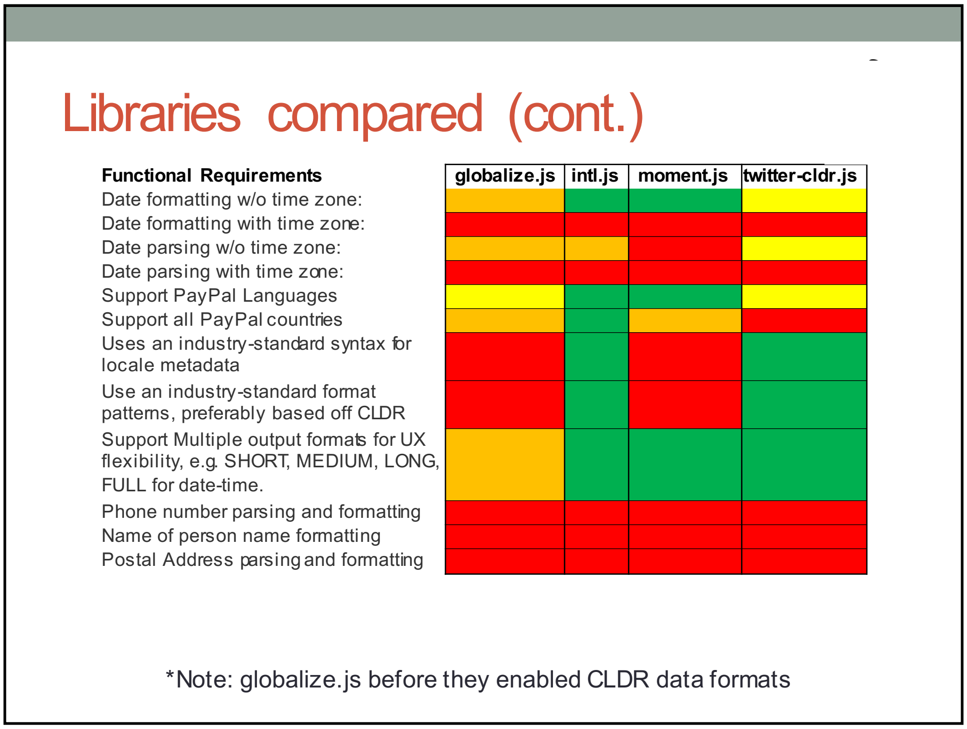
- JavaScript has thousands of open source solutions so surely this has been solved before?
- 2013 – looked at
  - Intl.js
  - Globalize.js
  - TwitterCLDR
  - CLDR.js
  - moment.js
  - Google closure



Libraries compared – all locales

Functional Requirements	globalize.js*	intl.js	moment.js	twitter-cldr.js
Locale resolution, given user session context and profile meta data	🟥	🟥	🟥	🟥
Number formatting into strings	🟨	🟩	🟥	🟨
Number parsing from strings	🟧	🟥	🟥	🟨
PayPal-specific Currency formatting into strings	🟥	🟨	🟥	🟧
Currency parsing from strings	🟥	🟥	🟥	🟥
Time Zone: Resolution from Olsen Timezones	🟥	🟥	🟨	🟥
Time Zone: Resolution from Timezone Offsets	🟥	🟥	🟧	🟥
Date formatting into strings, and Date parsing from strings, considering time zone	🟥	🟥	🟧	🟥
Time formatting into strings, and Time parsing from strings, considering time zone	🟥	🟥	🟧	🟥

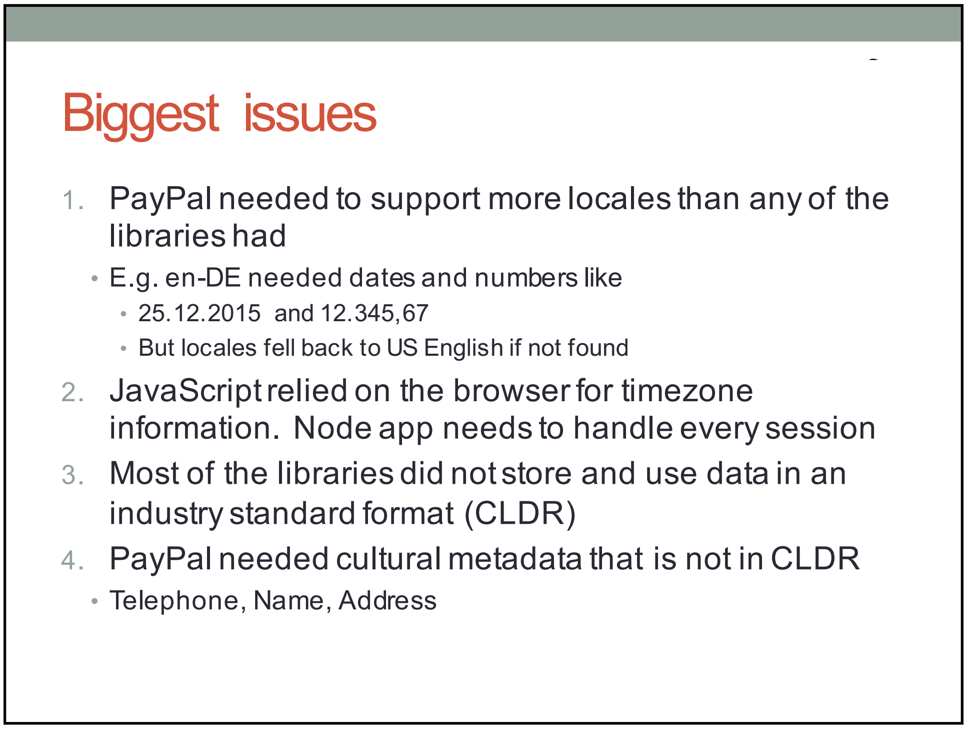
*Note: globalize.js before they enabled CLDR data formats



Libraries compared (cont.)

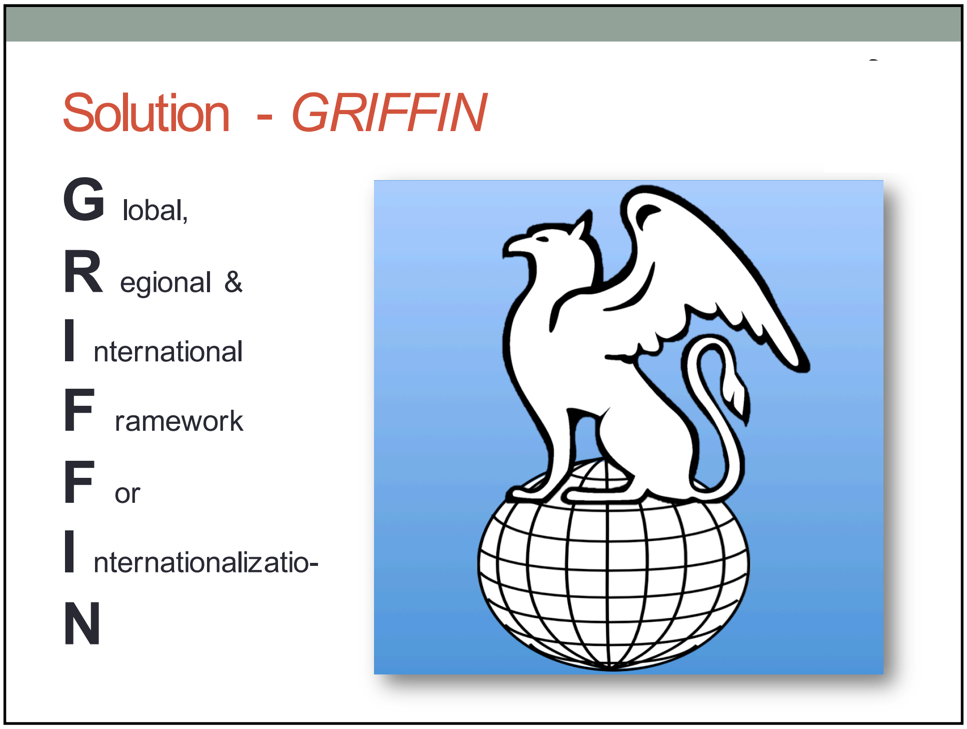
Functional Requirements	globalize.js	intl.js	moment.js	twitter-cldr.js
Date formatting w/o time zone:	🟧	🟩	🟩	🟨
Date formatting with time zone:	🟥	🟥	🟥	🟥
Date parsing w/o time zone:	🟧	🟧	🟥	🟨
Date parsing with time zone:	🟥	🟥	🟥	🟥
Support PayPal Languages	🟨	🟩	🟩	🟨
Support all PayPal countries	🟧	🟩	🟧	🟥
Uses an industry-standard syntax for locale metadata	🟥	🟩	🟥	🟩
Use an industry-standard format patterns, preferably based off CLDR	🟥	🟩	🟥	🟩
Support Multiple output formats for UX flexibility, e.g. SHORT, MEDIUM, LONG, FULL for date-time.	🟧	🟩	🟩	🟩
Phone number parsing and formatting	🟥	🟥	🟥	🟥
Name of person name formatting	🟥	🟥	🟥	🟥
Postal Address parsing and formatting	🟥	🟥	🟥	🟥

*Note: globalize.js before they enabled CLDR data formats



Biggest issues

1. PayPal needed to support more locales than any of the libraries had
   - E.g. en-DE needed dates and numbers like
     - 25.12.2015 and 12.345,67
     - But locales fell back to US English if not found
2. JavaScript relied on the browser for timezone information. Node app needs to handle every session
3. Most of the libraries did not store and use data in an industry standard format (CLDR)
4. PayPal needed cultural metadata that is not in CLDR
   - Telephone, Name, Address



Solution - *GRIFFIN*

G lobal,

R egional &

I nternational

F ramework

F or

I nternationalizatio-

N

Solution - GRIFFIN

PayPal i18n framework for node.js

CLDR:

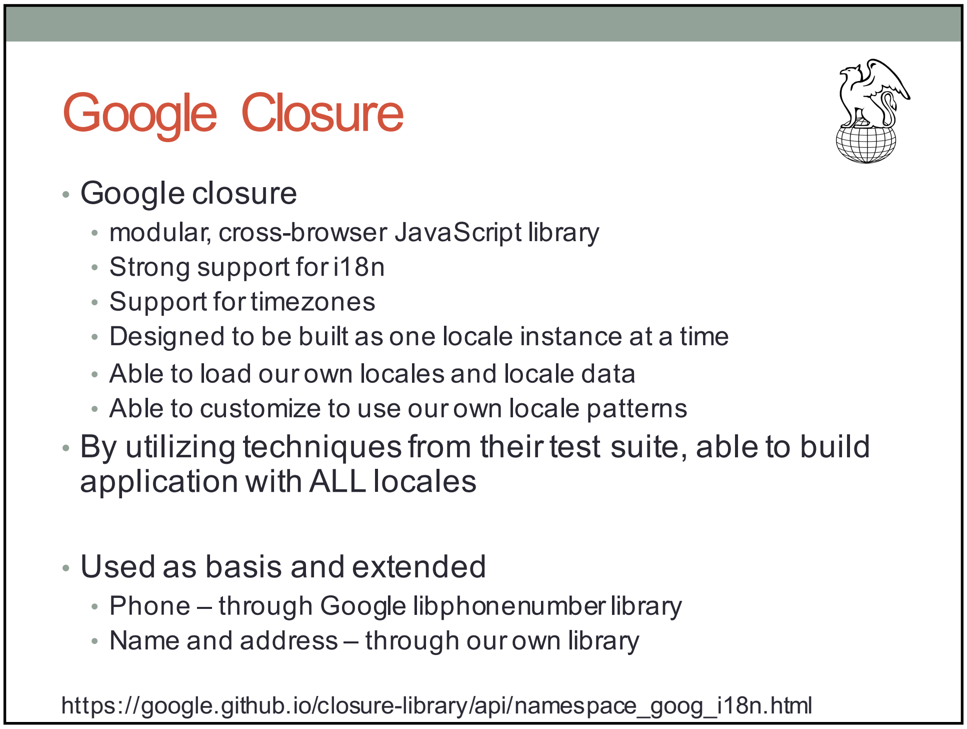
- Uses CLDR but had to override parts of CLDR to suit PayPal country and language requirements.
- Date/Time/Number/Currency Formats

Added beyond CLDR:

- UI Text Strings (for input guidance)
- Name Support
- Address Format
- Phone Number Support

Technical Challenges

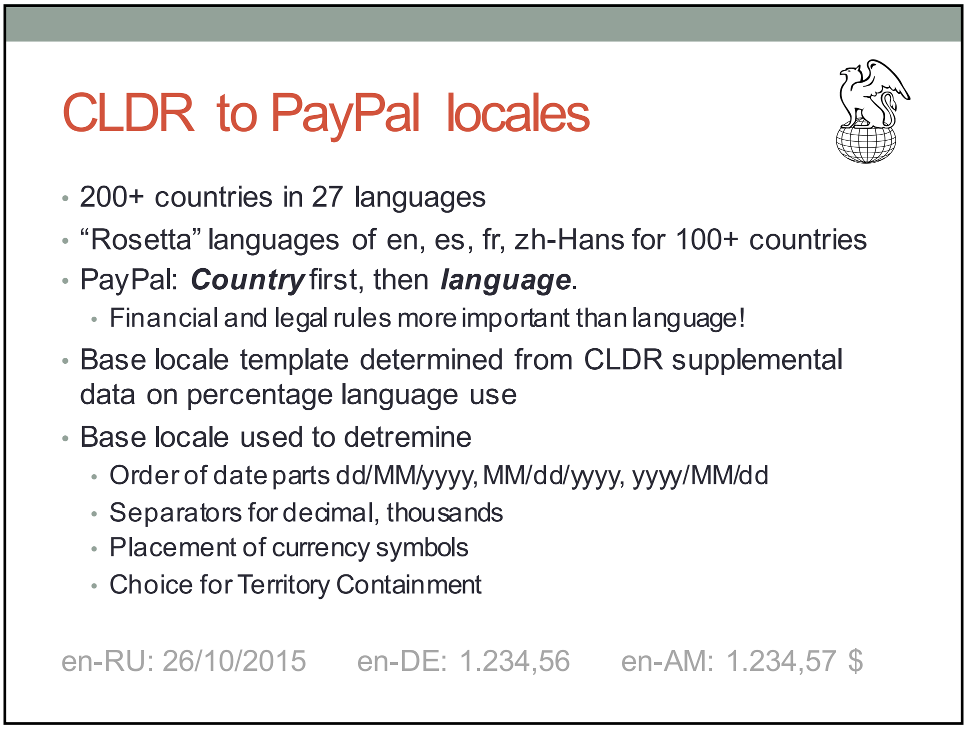
- Google Closure
- CLDR to PayPal locales
- Territory Containment and Locale Overrides
- Language+Country
- Size of data



Google Closure

- Google closure
  - modular, cross-browser JavaScript library
  - Strong support for i18n
  - Support for timezones
  - Designed to be built as one locale instance at a time
  - Able to load our own locales and locale data
  - Able to customize to use our own locale patterns
- By utilizing techniques from their test suite, able to build application with ALL locales
- Used as basis and extended
  - Phone – through Google libphonenumber library
  - Name and address – through our own library

https://google.github.io/closure-library/api/namespace_goog_i18n.html



CLDR to PayPal locales

- 200+ countries in 27 languages
- “Rosetta” languages of en, es, fr, zh-Hans for 100+ countries
- PayPal: ***Country*** first, then ***language***.
  - Financial and legal rules more important than language!
- Base locale template determined from CLDR supplemental data on percentage language use
- Base locale used to detremine
  - Order of date parts dd/MM/yyyy, MM/dd/yyyy, yyyy/MM/dd
  - Separators for decimal, thousands
  - Placement of currency symbols
  - Choice for Territory Containment

en-RU: 26/10/2015 en-DE: 1.234,56 en-AM: 1.234,57 $

Territory Containment

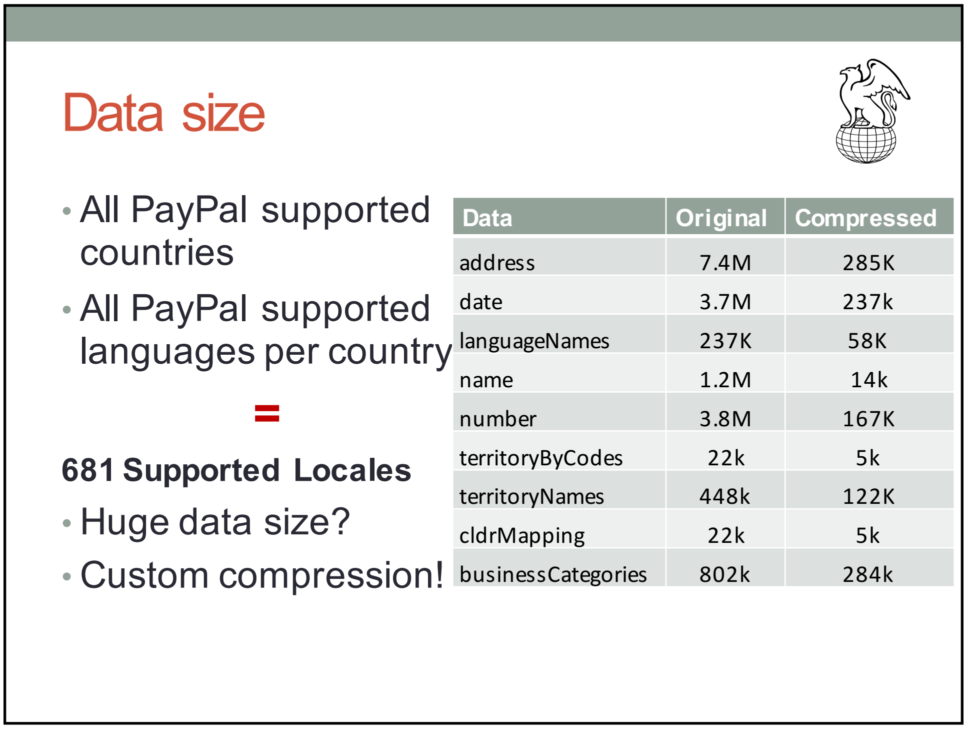
- CLDR Supplemental Data: UN M.49
  - World [Asia [Southern Asia [Vietnam
- Used for fall-back determinations
- Usual: en-XX not found? Use en, which looks like en-US
- Common fallback creates offensive or confusing formats

- CLDR only had
  en-001 (world), en-150 (europe), es-419 (latam)
- Created our own Territory fallbacks
  - based on predominant locale signature of:
  `date order + 12/24hr + number separators + currency placement`

http://www.unicode.org/cldr/charts/latest/supplemental/territory_containment_un_m_49.html

Language + Country

- Since PayPal needs to consider Country first, we cannot have any "generic" locale based only on language
- Locale is always a tuple of
  - `country`, for legal jurisdiction, local formats source
  - `locale` (language-country) for linguistic source
- For portability between legacy applications, web services, and JavaScript
  - Map between PayPal locale and
    - Legacy
    - CLDR
- Internally, a locale is composed of
  - closest CLDR language for all labels and string values
  - Territory container locale for date order, time, numbers



Data size

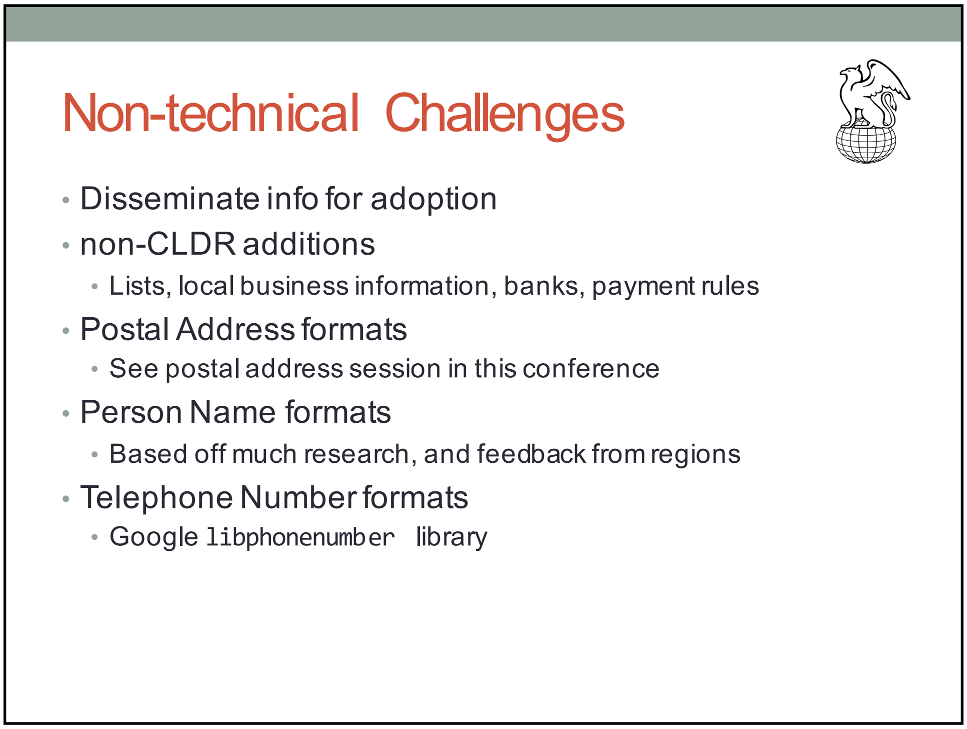
- All PayPal supported countries
- All PayPal supported languages per country

=

681 Supported Locales

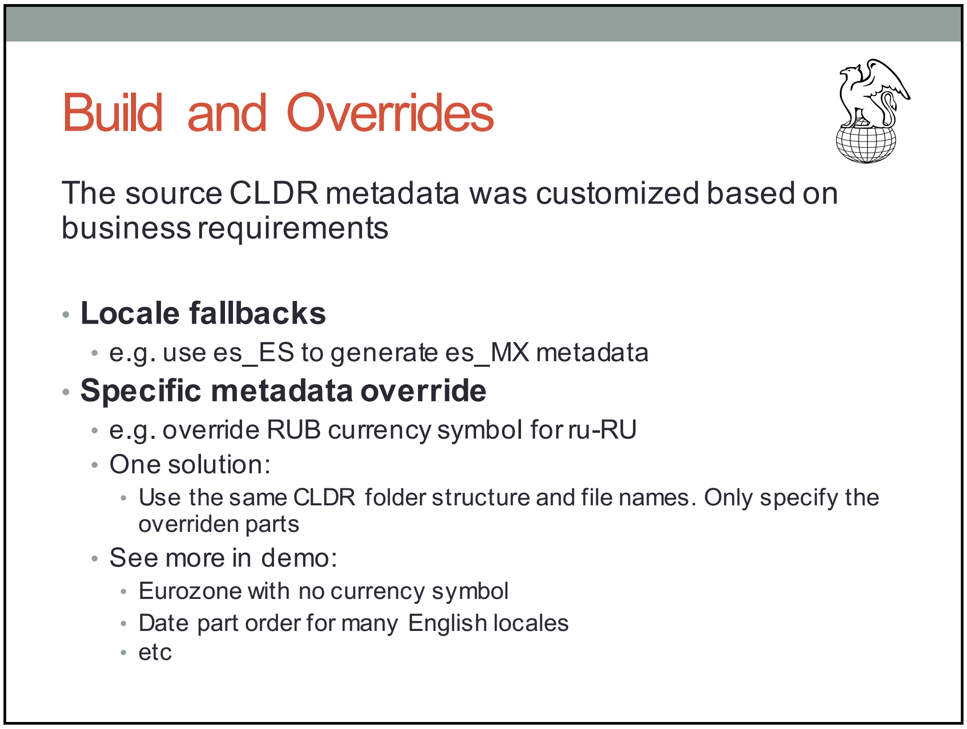
- Huge data size?
- Custom compression!

Data	Original	Compressed
address	7.4M	285K
date	3.7M	237k
languageNames	237K	58K
name	1.2M	14k
number	3.8M	167K
territoryByCodes	22k	5k
territoryNames	448k	122K
cldrMapping	22k	5k
businessCategories	802k	284k



Non-technical Challenges

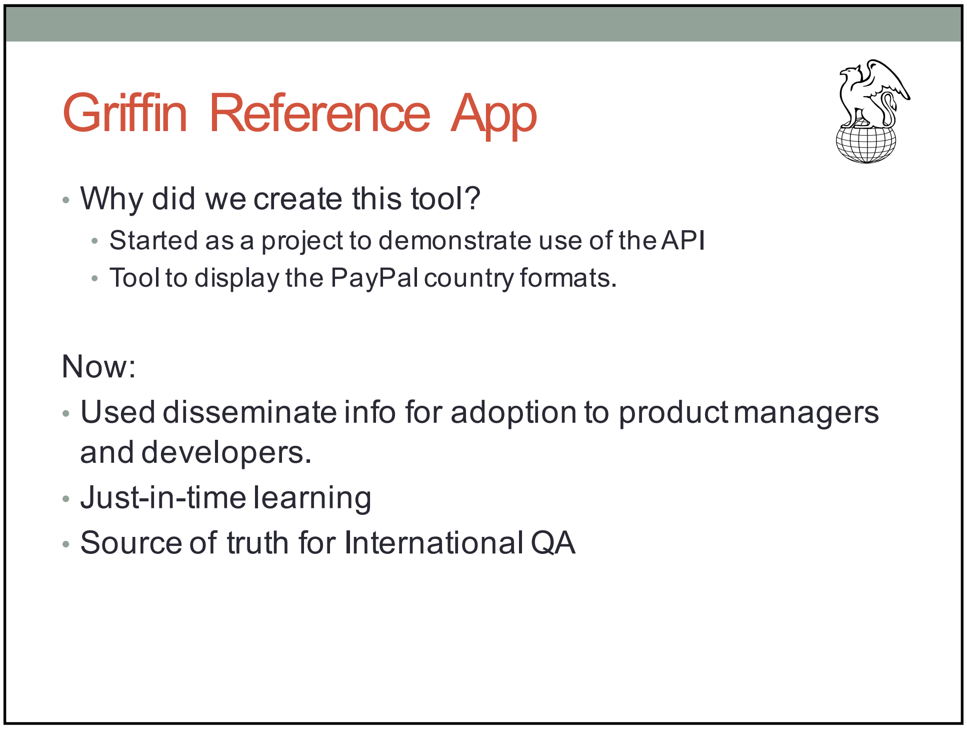
- Disseminate info for adoption
- non-CLDR additions
  - Lists, local business information, banks, payment rules
- Postal Address formats
  - See postal address session in this conference
- Person Name formats
  - Based off much research, and feedback from regions
- Telephone Number formats
  - Google `libphonenumber` library



Build and Overrides

The source CLDR metadata was customized based on business requirements

- **Locale fallbacks**
  - e.g. use es_ES to generate es_MX metadata
- **Specific metadata override**
  - e.g. override RUB currency symbol for ru-RU
  - One solution:
    - Use the same CLDR folder structure and file names. Only specify the overriden parts
  - See more in demo:
    - Eurozone with no currency symbol
    - Date part order for many English locales
    - etc



Griffin Reference App

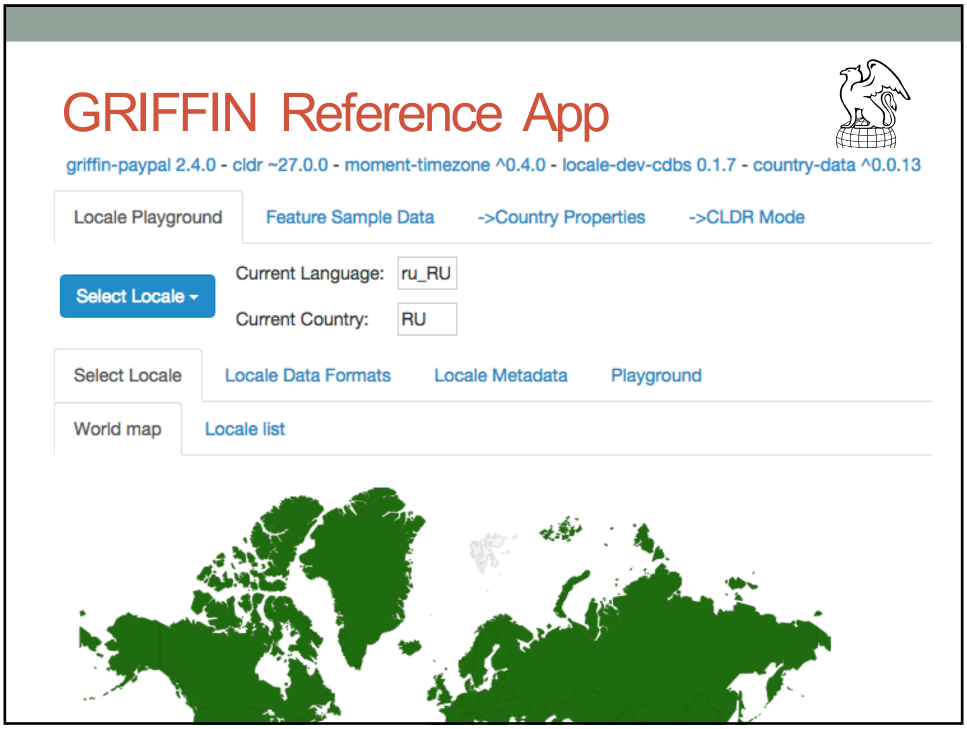
- Why did we create this tool?
  - Started as a project to demonstrate use of the API
  - Tool to display the PayPal country formats.

Now:

- Used disseminate info for adoption to product managers and developers.
- Just-in-time learning
- Source of truth for International QA

DEMO

Griffin Reference App

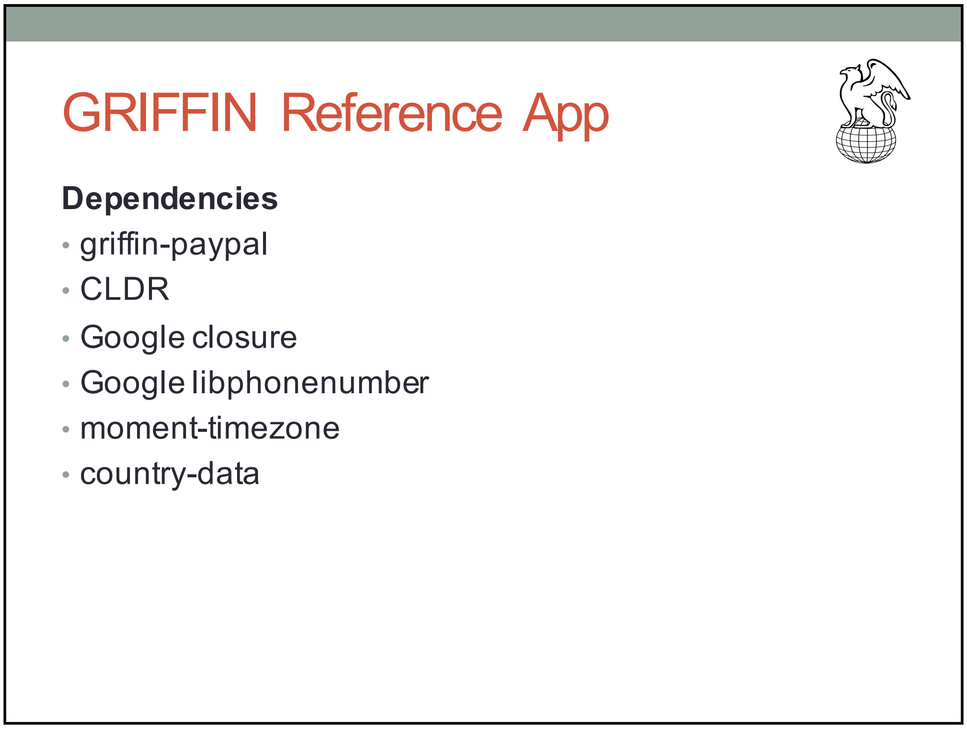


GRIFFIN Reference App

griffin-paypal 2.4.0 - cldr ~27.0.0 - moment-timezone ^0.4.0 - locale-dev-cdbs 0.1.7 - country-data ^0.0.13

Current Language: ru_RU

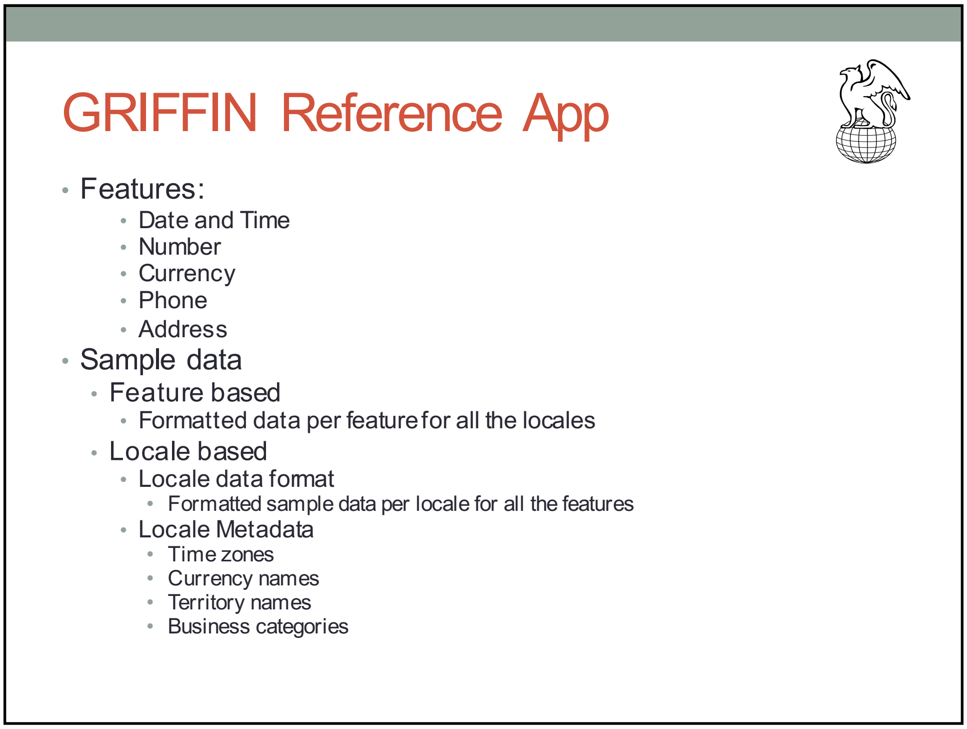
Current Country: RU



GRIFFIN Reference App

Dependencies

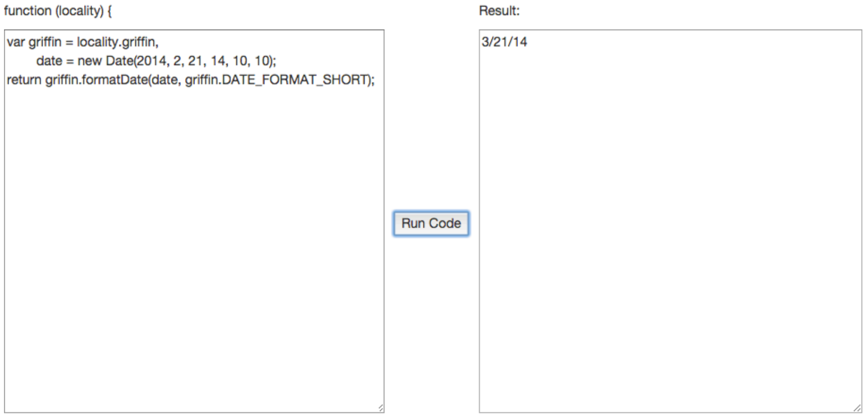
- griffin-paypal
- CLDR
- Google closure
- Google libphonenumber
- moment-timezone
- country-data



GRIFFIN Reference App

- Features:
    - Date and Time
    - Number
    - Currency
    - Phone
    - Address
- Sample data
    - Feature based
        - Formatted data per feature for all the locales
    - Locale based
        - Locale data format
            - Formatted sample data per locale for all the features
        - Locale Metadata
            - Time zones
            - Currency names
            - Territory names
            - Business categories

Playground



function (locality) {

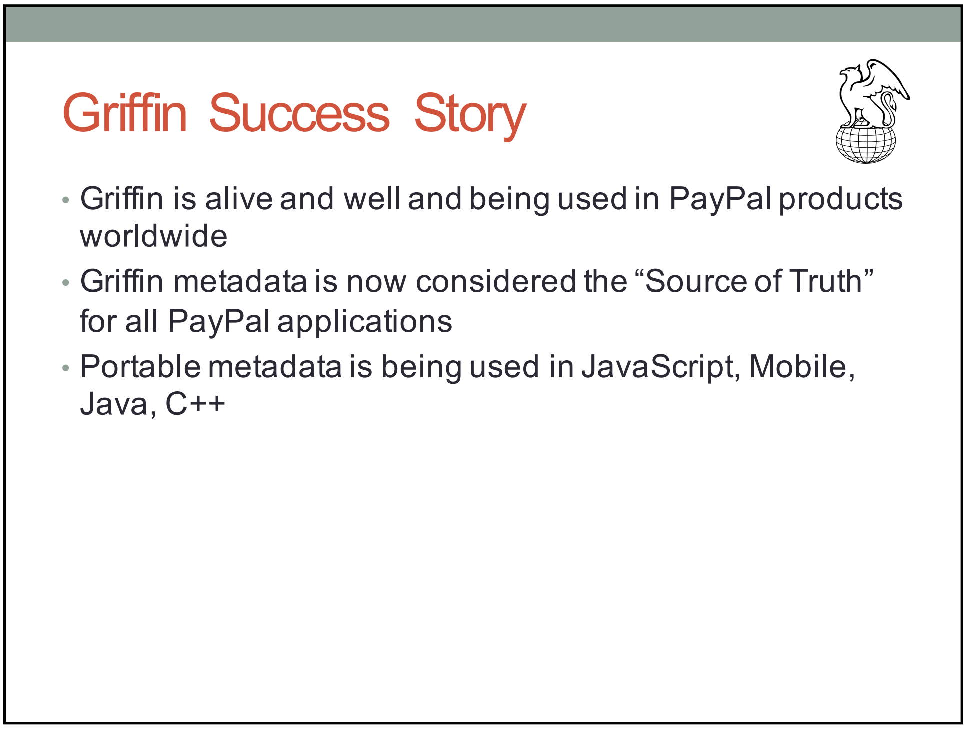
```
var griffin = locality.griffin,
        date = new Date(2014, 2, 21, 14, 10, 10);
return griffin.formatDate(date, griffin.DATE_FORMAT_SHORT);
```

Result:

```
3/21/14
```

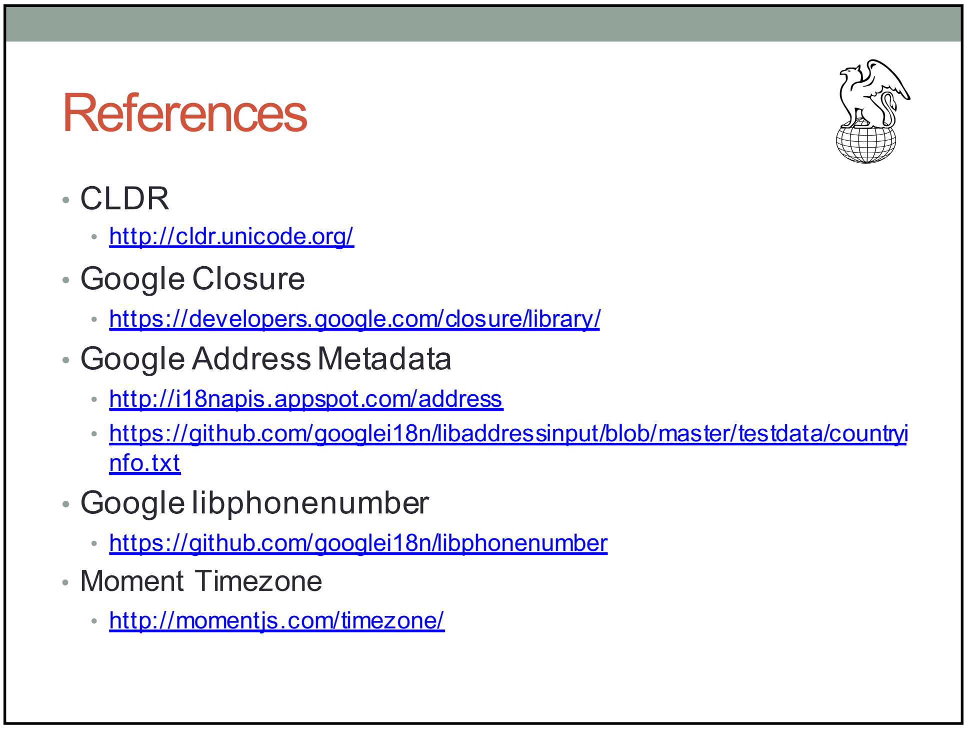
Lessons Learned

- Today: globalize.js is much more powerful. Would start there if we did it again.
- Re-publish our internal metadata in LDML/CLDR schemas as much as possible. We are returning in closure.js formats now which have to be rearranged to be useful
- Submit more tickets, with background, to CLDR
- Think more about business/regional/user feedback



Griffin Success Story

- Griffin is alive and well and being used in PayPal products worldwide
- Griffin metadata is now considered the “Source of Truth” for all PayPal applications
- Portable metadata is being used in JavaScript, Mobile, Java, C++



References

- CLDR
  - http://cldr.unicode.org/
- Google Closure
  - https://developers.google.com/closure/library/
- Google Address Metadata
  - http://i18napis.appspot.com/address
  - https://github.com/googlei18n/libaddressinput/blob/master/testdata/countryinfo.txt
- Google libphonenumber
  - https://github.com/googlei18n/libphonenumber
- Moment Timezone
  - http://momentjs.com/timezone/