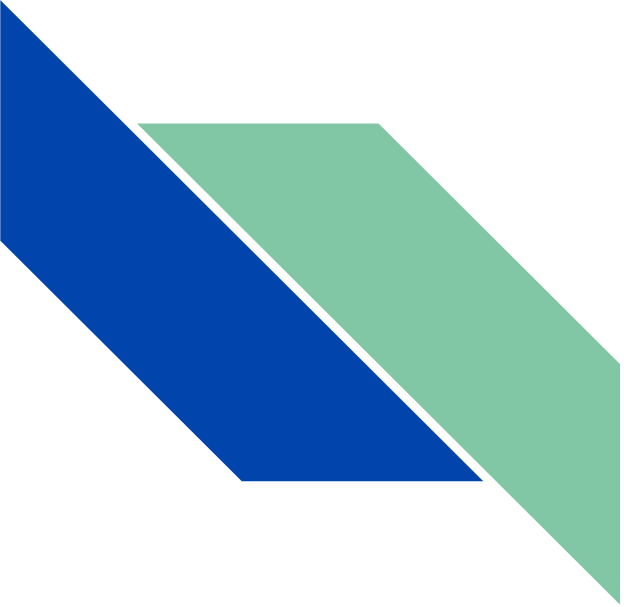


Internationalisation(i18n) & Localisation(l10n)

Mayank Kumar

Agenda

- Multilingual Softwares/Apps
- Internationalisation (i18n)
- Localisation (l10n)
- Need for i18n and l10n
- The Process
- Challenges & Complexity
- RTL
- Browser Support

Is My App Ready?

Multilingual Softwares

- Softwares/Apps rendering in multiple languages
- Future of application development
- Require 2 phases: i18n and l10n
- Globalization is the end goal as every business needs a global expansion through their platforms

Internationalisation (i18n)

- i*<18 letters>*n
- Process of making l10n possible. Capability to adapt different locales.
- Backbone of app translation.
- No runtime or deployment-time changes required.
- Text, Images, Currency, Numerics, Punctuations.

Localisation (l10n)

- l*<10 letters>*n
- Translation to different locales.
- Target based localisation.
- Locale product translations and testing is required.
- Not everything can be localised.

Need for i18n and l10n

- Global market footprint.
- Customer satisfaction.
- Sales growth.
- Avoiding cultural conflicts.
- Reduce “*Time to market*” for new market.

The Process

- Move all your textual content to key-value pair configs(.json/.properties).

```
<div>Hello World</div>
```

```
{
    "HELLO_CONTENT": "Hello World"
}
```

```
{
    "HELLO_CONTENT": "Hallo Welt"
}
```

```
<div>{CONFIG.HELLO_CONTENT}</div>
```

- Do contextual translation of the configs.
- Take user's language preference and pick the right config.

The Process (contd.)

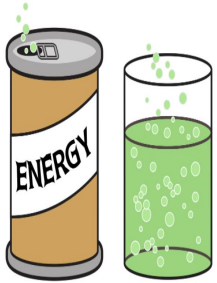
- Change number, currency formatting.

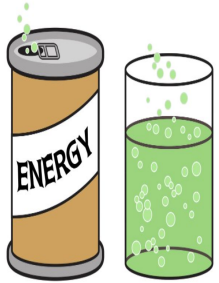
```
const number = 123456.789;
console.log(new Intl.NumberFormat('de-DE', { style: 'currency', currency: 'EUR' }).format(number));
// expected output: "123.456,79 €"
// the Japanese yen doesn't use a minor unit
console.log(new Intl.NumberFormat('ja-JP', { style: 'currency', currency: 'JPY' }).format(number));
// expected output: "￥123,457"
// limit to three significant digits
console.log(new Intl.NumberFormat('en-IN', { maximumSignificantDigits: 3 }).format(number));
// expected output: "1,23,000"
```

- Conditional rendering based on selected language like *Translate to English* action.

Challenges & Complexity

- Implementing i18n in existing app.
- Contextual translation.
- Translation cost.
- Localisation testing. Accuracy matters.
- Overhead in case of partial translation.


ENERGY


ENERGY

Right-to-Left

- Rendering of the RTL languages such as Arabic, Urdu, Hebrew, Farsi.
- Layout toggling.
- Input language change.
- Text representation.
- Image transitions and slide shows.

Browser Support

- HTML attribute: ***dir=”rtl”***
- HTML tag: <bdi><bdi>
- CSS: ***direction=”rtl”***
- JavaScript built-in function support

Other things to take care

- Translate the text as a whole, do not translate partially.
- Don’t just rely on language code, consider country codes as well (`en_US, en_GB`).
- Do not design your app based on string length.

Thank You
Let’s go global


www.linkedin.com/in/mayank-kumar-in