API Product Strategy for scaling different product lines

The journey of REST APIs

Adrita Bhor • 25th Sept 2018
Behind every successful product at PayPal there is an API
THE BUSINESS OF APIS

What is it?
Build a re-usable platform for exchanging data to power innovative applications.

Why do APIs need product managers?
• Rationalize requirements
• API Design
• Gather feedback
• KPIs and measurement
• Adoption

Who is it targeted towards?
• Merchants, partners, marketplaces, developers
• Level of understanding varies
REQUIREMENTS

• Data – customer first
  • Developers
  • Merchants
  • Partners
  • Marketplaces

• Rationalizing across multiple groups in the organization

• What to build – how granular

• Basic building blocks of every domain

• Single integration platform – integrations that evolve as business grows
API DESIGN

• Keeping it SIMPLE

• Humanizing APIs – journey from /payment, -> ebay API -> /orders

• Consistency

• API Governance

• Style guides, standards

• Constant feedback loop with external and internal integrators

• Versioning and deprecation
## Strategy with least bottlenecks

<table>
<thead>
<tr>
<th>Merchant Engagement &amp; Onboarding</th>
<th>Developer Portal</th>
<th>Partner Onboarding</th>
<th>PayPal.com</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Security and Identity</strong></td>
<td></td>
<td>Identity Access Token (/v1/oauth2/token)</td>
<td></td>
</tr>
<tr>
<td><strong>API Products</strong></td>
<td>Checkout</td>
<td>Subscriptions</td>
<td>Invoicing</td>
</tr>
<tr>
<td><strong>Payment Operations</strong></td>
<td>Authorize</td>
<td>Capture</td>
<td>Refund</td>
</tr>
<tr>
<td><strong>Search and Reporting</strong></td>
<td>List Activities</td>
<td>Transaction Reporting (/v1/reporting/transactions)</td>
<td></td>
</tr>
<tr>
<td><strong>Backoffice APIs</strong></td>
<td></td>
<td>Disputes</td>
<td></td>
</tr>
</tbody>
</table>
SIMPLICITY OF ORDER API
Integration starts with just amount field

Basic API call

```
{
  "intent": "CAPTURE",
  "purchase_units": [{
    "amount": {
      "currency_code": "USD",
      "value": "100.00"
    }
  }]
}
```

+ Mix and Match objects per use case

- **purchase_unit.items**: Merchants can provide additional item information

- **purchase_unit.shipping_address**: Merchants can send the address they collected or get the payer address upon buyer approval

- **purchase_unit.amount.breakdown**: Merchants can optionally choose to provide breakdown of amounts (previously amount.details)

- **application_context**: Allows API caller to customize the buyer experience (similar to the way Payments Experience API did historically)

- **payer**
  When passed within the request this serves to prefill the Hermes Guest experience. Upon buyer approval this contains details about the Buyer.

- **payee**
  For 3rd party use cases API callers can also specify the payee.
API LAUNCH FRAMEWORK

• API Specification / Postman Scripts
• Integration Guides
• Sandbox Environment
• Training Programs – Internal and External
• Product FAQ, Datasheet and Videos
• PR and Communication
• Developer Portal Updates

EAT. | SLEEP. | LAUNCH.
API ECOSYSTEM

Things that make an API successful

- Developer Documentation Accuracy
- Negative Testing, Sandbox
- SDKs
- Debugging & Monitoring
- Notifications
- API Governance [ML5] *

* ML – Maturity Level of API [1-5]
SUMMARY - LESSONS LEARNED

• Treat APIs as products and understand customer needs first

• Work from a minimal API structure that can grow over time

• Make your default design the least constrained

• API have experience too – the “integration experience” - APIs cannot be successful without the ecosystem

• Versioning and deprecation – Have a clear deprecation plan for your APIs to keep the overall offering simple.