Twitch Developer Library Requirements

Introduction

The Twitch Developer Library (TDL) is a content library designed to facilitate developer interaction with the Twitch platform. The TDL enables developers to use the Twitch platform more efficiently by providing relevant support tools, technical documentation, and help articles.

To determine the feature requirements for this project two phases of research were conducted. The first phase consisted of interviewing potential users of the TDL. Throughout the course of a week I interviewed four senior game developers, three junior-level game developers and SDE, two producers, and one designer all within AGS. These interviews helped determine the key function a developer library servers for each position and ultimately who the key users are. The next research phase consisted of competitive analysis of other open-access developer portals. I identified seven competitors and documented the core features and strengths and weaknesses of the portals.

There are three main roles that makeup game teams and each role has a different reason for visiting a developer portal. Game developers visit developer portals to confirm game design requirements are possible on the game platform. Developers use the developer portal to look up API outputs to determine the tech requirements, analyze system capabilities, and generally understand the key tech involved on the platform. Game designers generally use developer portals to check system requirements, understand analytic metrics available, and understand the available tools on the platform. Game producers have different use cases, naming checking software release cadences and software roadmaps.

The combination of the user interviews and competitive analysis determined the target user of the TDL and the TDL product goal. The target user of the TDL is an experienced AAA game developer with the goal of integrating his or her game with Twitch in order to promote the discovery of the AAA title. Game developers are the target users of the TDL because of the frequency with which they will use the developer portal, as opposed to other game roles such as designers and producers who visit developer portals at a lower frequency.

This document outlines the specific needs of game developers by outlining how the key findings correlate with the key features that are necessary to the target user.

Overview

<u>Key Interview Findings</u>

Key User Groups

1. Game Developers
    a. Customers who create intricate features for work-related projects.
    b. Primarily search and scan documentation on DL.
    c. Frequent forums like StackOverflow and prefer finding answers through forums.
2. Game Designers
    a. Customers who need to understand the system requirements of Twitch and integration tools Twitch provides for work-related projects.
    b. Scan and read guides and tool pages during the onboarding process for a new platform.
3. Game Producers
    a. Customers who manage the game production process and monitor the release cadences of Twitch software.
    b. Navigate directly to the landing page of a developer portal or relevant article to find software update information.

Use Cases

1. First Time Visitors
    a. Spend anywhere from 30-60 seconds searching for an answer on a page before leaving.
    b. Will spend time visiting other pages on the site from the internal navigation to better understand the key technical requirements for faster onboarding.
    c. Use “how to’s” and sample code for onboarding with new personal or professional projects.

2. Returning Visitors
    a. Spend several seconds scanning the “above the fold” on a page before leaving in order to answer questions quickly.
    b. Once the customer has found the answer, he or she will leave the site.

User Stories

I find out about dev.twitch.tv from online news articles and word of mouth.

I use Google to search a specific topic and navigate to the Twitch developer portal.

As a new visitor I arrive to the site via a specific topic page then navigate to the home page.

As an experienced game developer, I scan sample code to find the information I need.

As a new visitor I have confidence I found the right information in my search because I’m able to review the most popular features Twitch provides and see the high level navigation on the landing page.

As a producer I visit the home page to find information on Twitch software release cadences.

As a designer I visit the home page to understand the different topics and tools Twitch provides and to read article pages about new features.

I explore the site by scanning documentation to find subtopics relevant to features I’m designing.

As a junior developer I use the QA forum whenever I have a question that can’t be answered by a quick scan of the documentation.

As a return visitor I navigate to the Twitch developer blog to stay informed on new announcements.

User Scenario

An AAA game studio wants to compete in the in-app purchase market for PC games. The game producer meets with the leaders of this studio and decides to make this PC game available on all PC game marketplaces. Additionally the studio wants to enable more player engagement with the game through platform and marketplace integration. These business requirements are handed down to the game design team who decided to create this game for the Twitch platform, among others. These designers visit the Twitch developer portal to see the integration tools available and use these insights to write the design requirements. The design team then hands off the requirements to the developer team. A group of experienced developers meet to determine if these design requirements fit the technical requirements of the Twitch platform. These developers visit the Twitch portal to scan the technical capabilities of the platform. Once each requirement is green-lit, the development management team assigns feature development to a junior developer. This junior developer is responsible for the implementation of each feature assigned to him or her. In order to start development each junior developer conducts in-depth research of each feature by reading documentation and code samples on the Twitch developer portal. Throughout the development process the junior developer will return to the developer portal as a reference.

User Persona

Nathan, Experienced AAA Game Developer

Nathan is a game developer with 7 years of experience in the game industry. He started as a game developer out of college and has worked at several companies including Blizzard and EA. Nathan has been at EA for 2 years and recently transferred internally to a new team working on an AAA PC game to be released in 2018. Nathan is now working with twenty other game developers and is currently in the pre production phases of the project. Last week the game designers released the first draft of the player experience goals to the developer team. The head of the EA PC gaming division has made it clear to the developer team that the game will be released and sold on all the popular distribution channels including Steam, Twitch, and Desura. With these experience and business requirements in mind, Nathan and the other programmers must determine the feasibility of the requirements and the key tech requirements each distribution platform has. Nathan is expected to craft a report on the feasibility and tech requirements necessary for the game in the next couple weeks before game production begins.

Having worked on AAA PC titles before, Nathan is familiar with the distribution and API integration available on Steam and Desura. Twitch is an unfamiliar platform for Nathan so he navigates directly to the developer portal from a Google search to download the Twitch SDK. In order to understand the high level functionality of the Twitch SDK and APIs, he reads a short onboarding guide and then searches for the relevant API documentation. Nathan scans API documentation pages looking for JSON output types to determine the functionality each API would have in his game. Nathan sans each API documentation page on the Twitch portal and notes witch ones to include in his report. If Nathan ever needs to refer to the documentation page again he knows exactly which search terms to use. He will navigate to different Twitch API pages from Google and spend no more than a couple seconds on the page before leaving. From this point on, the Twitch developer portal is a mere reference for him and he will spend little time on it as he enters the production stage of the game.

User Experience Requirements

For the target user the most important requirements are:

1. Documentation with code samples
2. Search
3. Tutorials

These additional requirements are ranked in order of importance:

- Guides
- Reference Documentation (Class lists)
- Tools/ Article Pages
- Help/ FAQ
- Visuals
- News
- Release Cadences

Appendix

Twitch Competitive Analysis

For this analysis I focused on open access or open source developer portals that corresponded with a product or service, such as an API or SDK. The goal of the Twitch developer portal is to provide information on the integration of Twitch into games. I chose not to focus on platform developer portals such as Steamworks, Desura, Android, or IOS because these portals tend to be closed access and describe development platforms. However, I have linked to some of these platform portals as design references in the Appendix.

1. Twitch Competitors
    a. Twilio Documentation
        i. Twilio is an open-access portal focused on integrating Twilio cloud communication APIs with customer apps.
    b. Parse Documentation
        i. Parse is an open-access portal targeting the download and use of Parse cloud hosting APIs for all languages and platforms.
    c. Stripe Documentation
        i. Stripe is an open-access mobile payments portal targeted at answering questions via a documentation format.
    d. Zendesk Developers
        i. Zendesk is an open-access portal focused on integrating Zendesk customer service APIs, apps, and SDK with customer apps.
    e. Facebook Developers
        i. Facebook is an open-access portal focused on the integration of Facebook with customer apps.
    f. Google Maps
        i. Google Maps is an open-access portal, which is apart of the closed-access Google developer portal. Google Maps targets integrating Maps with customer apps.
    g. Unity Documentation
        i. Unity is an open-access portal designed to provide documentation for developers using the platform.

2. Competitor Core Features
    a. The highlighted core competitor features are the features important to the target user: experienced AAA game developers.

	Docs	Code Samples	Search	Tutorial	Ref	Guides	SDK↓	Tools	News	Release Cadence	Help
Twilio	√	√		√	√	√					√
Parse	√	√			√	√					
Stripe	√	√									√
Zendesk		√	√			√	√				√
Facebook	√	√	√			√	√	√	√	√	√
Maps	√	√	√	√	√	√				√	√
Oculus	√	√		√		√	√		√	√	
Unity	√	√	√								

3. Competitor Strengths & Weaknesses
    a. Twilio
        i. – Inline code samples missing
        ii. – Too many links to other pages
        iii. – Duplicate links within the body and on the right side navigation
    b. Parse
        i. + Clear organizational structure
        ii. + Clear call to action on landing page
        iii. + Defined topics
        iv. + Clear layout
    c. Stripe
        i. – Unclear call to action on landing page
        ii. + Clear navigation
        iii. + Clear content layout
    d. Zendesk
        i. – Unclear call to action on landing page
        ii. – Navigation back to landing page missing
        iii. + Clear content organization
        iv. + Easy to use QA support
    e. Facebook
        i. – Irrelevant call to action on landing page
        ii. – Information hidden in top navigation
        iii. + Clear content layout

f. Google Maps
    i. + Clear call to action on landing page
    ii. + Clear layout and navigation
g. Oculus
    i. – Unclear navigation to documentation
    ii. – Code samples are hard to read
    iii. + Clear call to action on landing page
    iv. + Clear content layout
h. Unity
    i. – References are automatically set to a Manual layout
    ii. – Users must navigate away to Unity Answers for support
    iii. + Landing page is documentation page
    iv. + Easy to copy and paste code

Design References

- Steamworks
- Desura Development
- Android Developers
- Apple Developers
- Heroku Dev Center