spined

Spined: Onboarding Guide

Table of Contents

1. Introduction

Welcome to Spined onboarding guide. This guide will help you get started with Spined and provide some examples of how to use it.

2. What is Spined ?

Spined is a Lazy Dynamic NFT Manager. Spined is a tool that brigdes the gap between traditional gaming platforms and the blockchain ecosystem. Its primary objective is to facilitate secure, efficient, and traceable transactions on the blockchain while offering a seamless experience for users. By connecting game artifacts with NFTs (Non-Fungible Tokens), it makes digital assets in games more valuable by ensuring their authenticity, ownership, and interoperability.

The "lazy" part of the manager helps game servers in two important ways:

- **Gamer Experience**: Game servers can run without needing gamers to approve transactions while they’re playing, which means gameplay isn’t interrupted.
- **Business Model Flexibility**: Spined separates updates in its database from updates on the blockchain. This lets game servers manage user actions without having to worry about the cost of blockchain transactions.

Spined operates with two main actors:

- **Spined API**: The Spined API is a server-side application that handles the creation, updating, and management of artifacts. It communicates with game servers and games, processes their requests, and updates its database accordingly. It is divided into two APIs:
  - The Game Server API, which requires an API key and should only be used by the game servers. This API is used to create, update, and manage artifacts.
  - A Public API, which is accessible to everyone and does not require authentication. This API allows anyone to fetch the attributes of artifacts, as well as the history of changes made to their attributes, and generate signed operations to update the blockchain representation of artifacts.
- **Smart Contract**: The Smart Contract resides on the blockchain. It’s responsible for the final validation and recording of artifact transactions.

These two actors work together to ensure the smooth operation of Spined, providing a seamless gaming experience.

3. Getting Started

3.1. Connecting to Spined

Each API of Spined has its own distinct URL.

The Game Server API requires an API key for authentication. Include this API key in the `x-api-key` header of your requests. This key is provided by our team and should be kept secure.

The Public API is open to everyone and does not require authentication. It allows anybody to use its endpoints.

> **NOTE** If you do not have the API URLs or API key, please contact our support team.

3.2. Smart Contracts

Spined uses Smart Contracts for classic web3 operations. These include minting, transferring, reading attributes, and checking ownership of artifacts.

By default, Spined handles ERC 721 or ERC 1155 tokens. However, it can be configured to handle other types of tokens, even custom ones that will be created specifically for your game. For more information, please contact our support team.

In Spined, we deploy a Smart Contract tailored to your project. However, if it suits your needs, it’s also possible to use a pre-existing Smart Contract that is shared between multiple projects. For more information, please contact our support team.

4. Available Actions

This section highlights the available actions in Spined API. For more details on the parameters and their formats, please refer to the API documentation.

> **NOTE** The API documentation is available at https://spined.tech/.

4.1. Creating New Artifacts

To create new artifacts, you can use the `mintArtifact` API endpoint. This requires the Contract ID of the artifact, the owner's address, and the amount of artifacts to create.

Alternatively, if you decide to handle the minting of artifacts yourself, or simply want to import existing artifacts, you can use the `registerArtifact` API endpoint. This endpoint allows you to add existing artifacts to Spined. To use this endpoint, you will need the Contract ID of the artifact you want to import.

4.2. Getting User Artifacts

The `getAddressArtifacts` API endpoint returns a list of artifact IDs owned by a user. Those ids can then be used to fetch the attributes of each artifact using the `getArtifact` API endpoint.

4.3. Getting Artifact Attributes

`getArtifact` API endpoint returns the attributes of an artifact. This can be used to fetch the attributes of an artifact owned by a user. Those attributes can then be used to determine the behavior of the artifact in the game, such as the color of a skin, its texture, etc.

4.4. Updating Artifacts Attributes

It is possible to update the attributes of an artifact using the `updateArtifactAttributes` API endpoint. This can be used in different ways, for example, to make a skin evolve after a specific action is performed by the player.

Those changes are only reflected in the database. To update the blockchain representation of the artifact, you will need to use the `signOperationsBatch` API endpoint. Refer to the Sign Artifact Operations action for more details.

4.5. Sign Artifacts Operations

The `signOperationsBatch` API endpoint generates manager-signed operations to update an artifact's blockchain representation based on the current database state. Those signed operations can then be sent to the blockchain by the game server or the user.

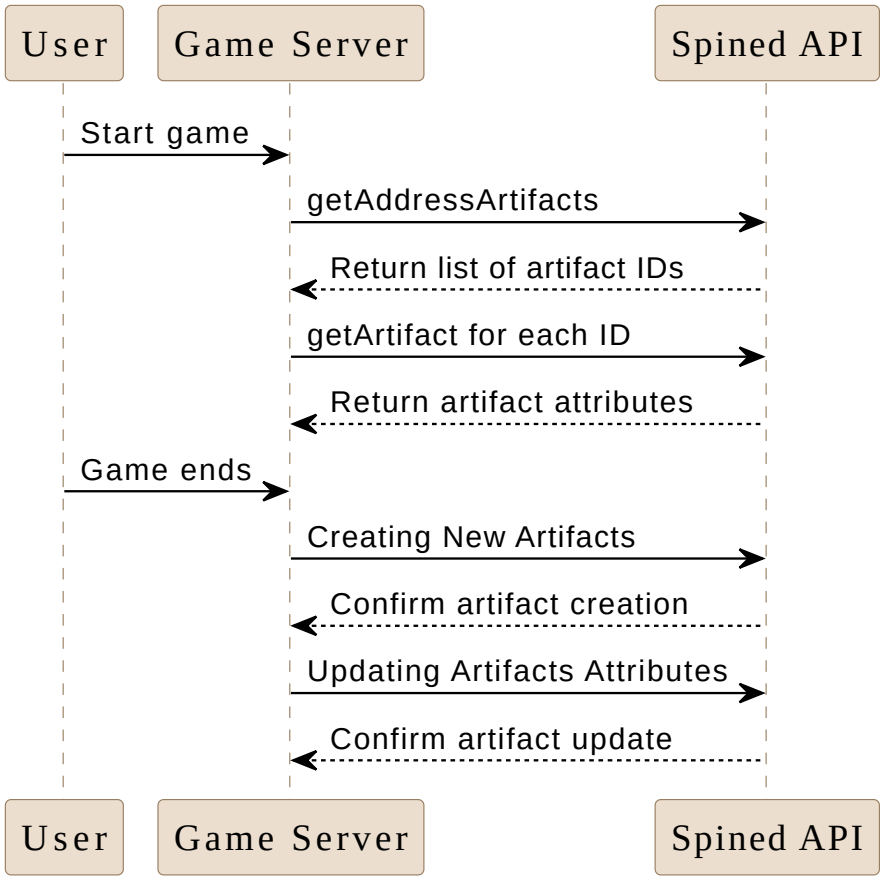
4.6. Getting Artifact History

The `getArtifactHistory` API endpoint returns all updates ever applied to the artifact's attributes. This can be used to track attribute changes over time, such as the evolution of a skin.

5. Use Cases

In this section, we will walk through an example of how Spined can be used in a game.

5.1. Classic Game



When a game starts up, it needs to fetch the player's artifacts inventory. This can be done in two ways:

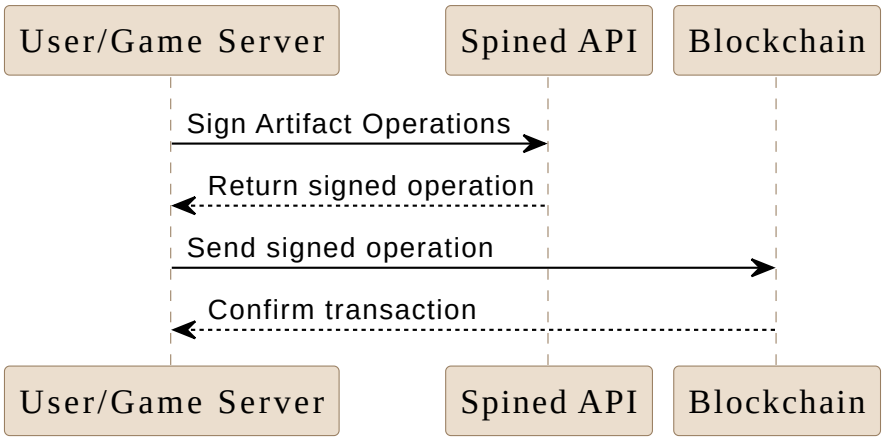
- The game server or the game itself can request the player's artifacts inventory using the `getAddressArtifacts` operation of Spined API. This operation returns a list of artifact IDs owned by the player.
- The attributes of each artifact can then be fetched using the `getArtifact` operation of Spined API.

> **NOTE** The data returned by the API is in raw format and may need to be deserialized before it can be used in the game.

Depending on the game, a player might acquire artifacts in various ways - through purchase, earning them, or as random drops. In these cases, the game server creates new artifacts using the Creating New Artifacts action.

Additionally, a player might already own an artifact that can "evolve" or change over time, such as a skin that alters after a certain action. In these cases, the artifact's attributes need to be updated. The game server can do this using the Updating Artifacts Attributes action.

5.2. Propagate artifact updates to the blockchain



Artifacts can be uploaded to the blockchain either by the game server or the user. They use the Sign Artifact Operations action to generate and sign the operation, then send it to the blockchain and pay the transaction fees.

This allows you to decide who is in charge of uploading artifacts to the blockchain and when, but also who pays for the transaction fees. This can be useful if you want to batch operations to reduce costs, or if you want to give the user more control over their artifacts and so, allow users to not use web3 features.

5.3. Buy/Sell my artifact

Artifacts in Spined are managed using Smart Contracts, specifically ERC 721 and ERC 1155 tokens. This allows users to buy or sell artifacts on any platform that supports these token standards. You could even build your own platform for this purpose.

1. **Selling an Artifact**: The owner can list their artifact on any platform that supports ERC 721 or ERC 1155 tokens. The transaction is completed on that platform.
2. **Buying an Artifact**: A buyer can purchase artifacts listed on these platforms. The platform handles the transfer of ownership.