Lesson 16 - NFTs / Metaplex

This week

Metaplex / NFTs
Anchor
Security

Other Languages

Seahorse

Docs

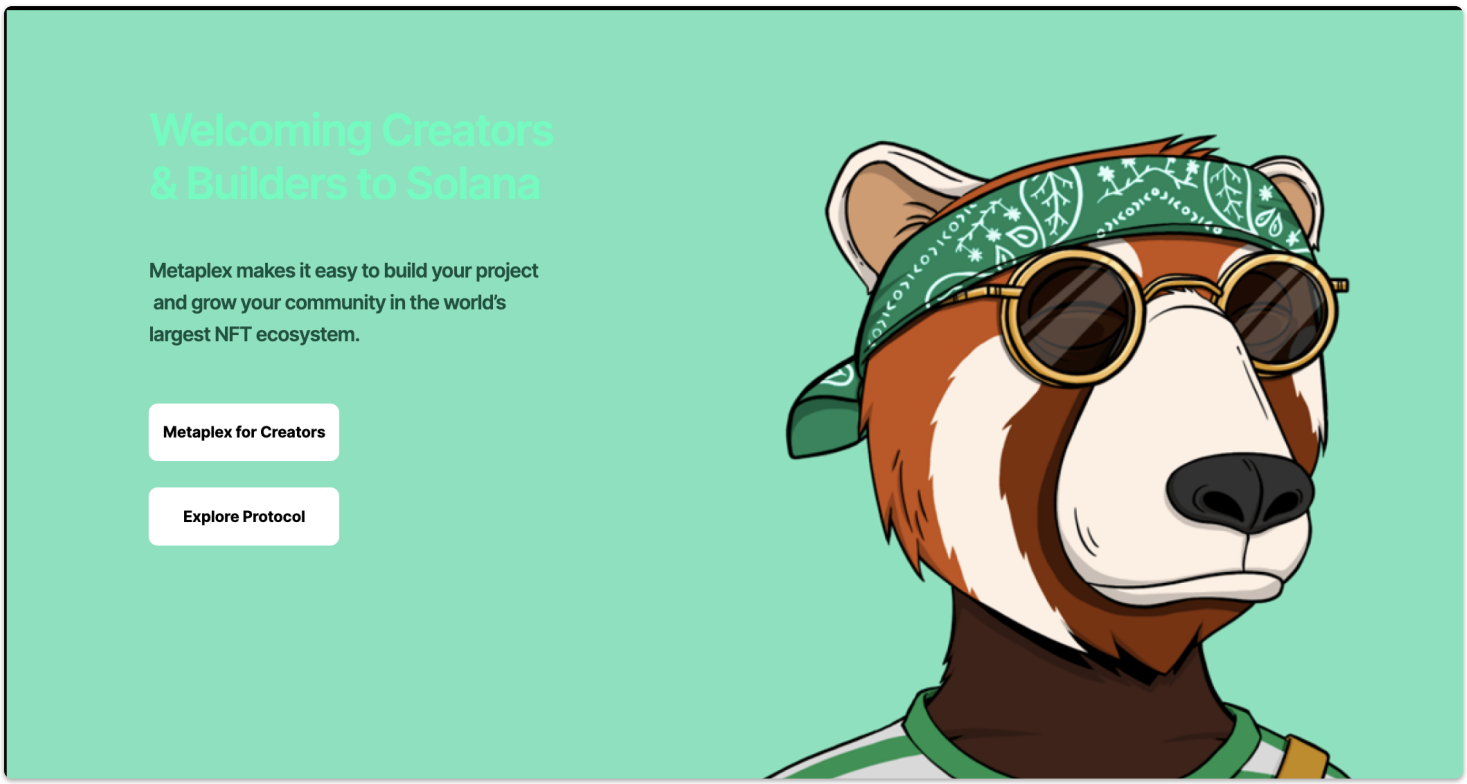
Seahorse lets you write Solana programs in Python. It is a community-led project built on Anchor.

Install with

```
cargo install seahorse-lang
```

It is also available in the Solana playground.

Metaplex



NFT Overview

NFTs are Non Fungible Tokens
They can be used for many things, but commonly they are used to tokenise or establish ownership of digital assets.

Although the standard was established in 2018, their popularity increased greatly in 2020 / 2021 and they became a mainstay of DeFi.

Non fungible means that each token is unique, unlike SOL for example for any one SOL is equivalent to any other.
An NFT therefore will have a means to identify it, we can then find who owns a particular NFT.

When used with digital assets, such as images the NFT will include a link to the asset.
Typically the assets are stored in decentralised storage systems such as Arweave, IPFS, NFT.Storage etc. The details of the location of the asset are often referred to as the metadata of the NFT.

More recently there has become a need for semi fungible tokens, these are SPL tokens with a supply greater than 1 but which has typical NFT attributes such as an image.

Metaplex overview

Metaplex is a framework for NFT launches, sales, auctions and partitioning that includes on-chain programs as well as client libraries. Some of the programs are not stable while others do not have a ready client.

Metaplex has 3 main projects

- Token Metadata - the NFT standard for Solana
- Candy Machine v3™ - a Profile Picture (PFP) focused tool that works like the gumball-style candy machines of old
- Auction House - a decentralized sales protocol for NFT marketplaces

Metaplex NFTs off-chain data can be stored on either Arweave, IPFS or AWS and multiple file formats are supported.
Wallets and websites that support Metaplex will be able to display all or some of these formats.

File Type	Extension
Audio	.mp3, .flac, .wav
Img	.png, .jpg, .gif
Movie	.mp4, .mov, .webm
VR	.gib
HTML	.html

Collections can be derived from existing collections.

Reasons to use Metaplex

A creator that is agnostic to the platform that will host NFTs, may consider Metaplex/Solana due to cheap gas costs.
The price of a single NFT mint varies due to the size of off-chain data and SOL price but will usually be under $3.

Metaplex architecture

Metaplex

A top level manager that knows the existence of other programs and interacts with them via CPI to accomplish necessary logic.

Token Metadata

This program allows decoration of SPL tokens with PDAs that store additional on-chain data and in them a pointer to the off-chain data.

Token data is split between:

- on-chain data
- off-chain data

On-chain data holds parameters such as:

- mint authority
- creator
- royalty splits
- link to off-chain metadata

The Non Fungible Standard from Metaplex.

See Docs

Off-chain data stores a pair of media file and associated JSON per NFT.
This JSON has anything that describes the individual parameters of this NFT.

▼ **Standard**

Field	Type	Description
name	string	Name of the asset.
symbol	string	Symbol of the asset.
description	string	Description of the asset.
image	string	URI pointing to the asset's logo.
animation_url	string	URI pointing to the asset's animation.
external_url	string	URI pointing to an external URL defining the asset — e.g. the game's main site.
attributes	array	Array of attributes defining the characteristics of the asset.
trait_type	string	The type of attribute.
value	string	The value for that attribute.

For example

```
{
  "name": "SolanaArtProject #1",
  "description": "Generative art on Solana.",
  "image": "https://arweave.net/26YdhY_eAzv26YdhY1uu9uiA3nmDZYwP8MwZAultcE?
ext=jpeg",
  "animation_url":
"https://arweave.net/ZAultcE_eAzv26YdhY1uu9uiA3nmDZYwP8MwuiA3nm?ext=glb",
  "external_url": "https://example.com",
  "attributes": [
    {
      "trait_type": "trait1",
      "value": "value1"
    },
    {
      "trait_type": "trait2",
      "value": "value2"
    }
  ],
  "properties": {
    "files": [
      {
        "uri": "https://www.arweave.net/abcd5678?ext=png",
        "type": "image/png"
      },
      {
        "uri": "https://watch.videodelivery.net/9876jkl",
        "type": "unknown",
        "cdn": true
      },
      {
        "uri": "https://www.arweave.net/efgh1234?ext=mp4",
        "type": "video/mp4"
      }
    ],
    "category": "video",

    // @deprecated
    // Do not use - may be removed in a future release.
    // Use on-chain data instead.
    "collection": {
      "name": "Solflare X NFT",
      "family": "Solflare"
    },

    // @deprecated
    // Do not use - may be removed in a future release.
```

```
    // Use on-chain data instead.
    "creators": [
      {
        "address": "xEtQ9Fpv62qdc1GYfpNReMasVTe9YW5bHJwfVKqo72u",
        "share": 100
      }
    ]
  }
}
```

Auction

This program supports the running of an auction, currently, only English auctions are supported. It has mechanisms for collecting bids (that can be any SPL token), cancelling bids, stating the winner and withdrawing funds.

The Auction PDA specifies parameters such as start and end time, authority or minimum bid price.

```
#[repr(C)]
#[derive(Clone, BorshSerialize, BorshDeserialize, PartialEq, Debug)]
pub struct AuctionData {
        pub authority: Pubkey,
        pub token_mint: Pubkey,
        pub last_bid: Option<UnixTimestamp>,
        pub ended_at: Option<UnixTimestamp>,
        pub end_auction_gap: Option<UnixTimestamp>,
        pub price_floor: PriceFloor,
    pub bid_state: BidState,
}
```

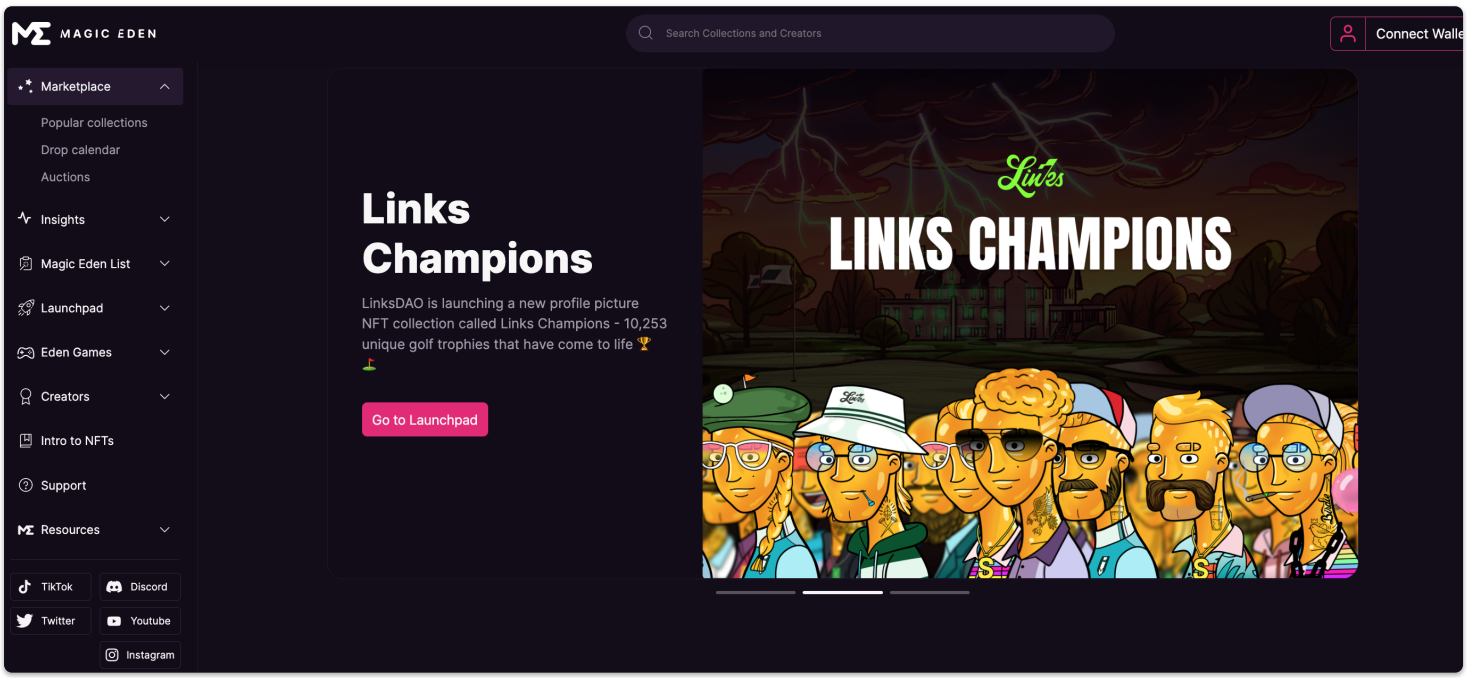
Token Vault

This program holds tokens being auctioned by the Auction program and allows for fractionalisation of an NFT. Partitioned NFT with a share like structure could be used so that multiple entities can own a single expensive asset.

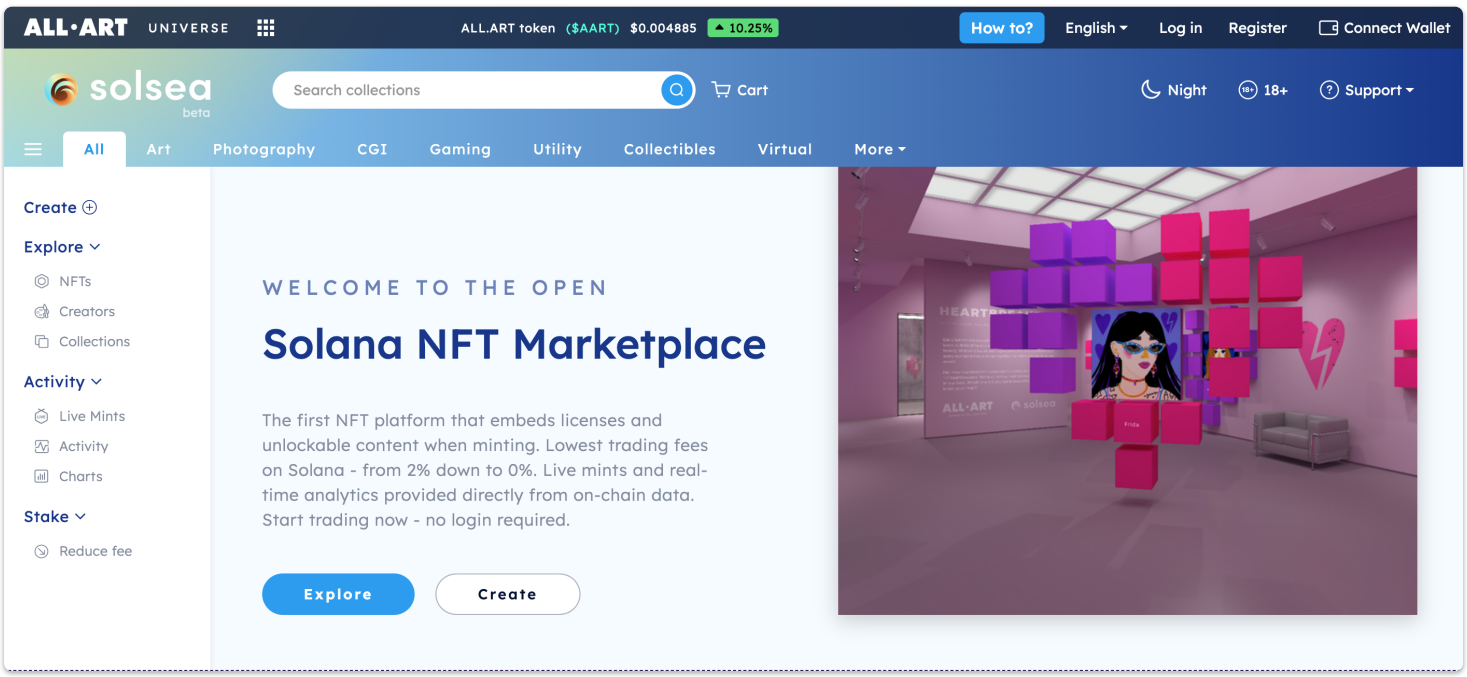
(NFT) Marketplaces

- Digital assets
- Can be fractionalised
- Can be used as collateral

Magic Eden



Solsea



Solanart

Solanart

Search collection, nft, user...

Collections | Sell items | Loans (New) | Launchpad | Connect wallet

Trade NFTs at ZERO cost

NEW Take out a loan in seconds. Read more

Buy or sell NFTs and save 100% in fees. The only zero-fee marketplace on Solana.

Explore collections | Sell my NFTs

DeGods (33.3%)
DeGod #2829
320

Candy machine

Candy machine has recently been upgraded.
See Docs
Features

- Accept payments in SOL, NFTs or any Solana token.
- Restrict your launch via start/end dates, mint limits, third party signers, etc.
- Protect your launch against bots via configurable bot taxes and gatekeepers like Captchas.
- Restrict minting to specific NFT/Token holders or to a curated list of wallets.
- Create multiple minting groups with different sets of rules.
- Reveal your NFTs after the launch whilst allowing your users to verify that information.

Candy Machine process flow

1. Create the machine and set up configuration.
2. Add the items we are minting
3. Mint the NFTs
4. Clean up

Candy Guards

There is the ability to add extra features to your candy machine by the use of 'guards'
Example guards

- Sol Payment: This guard ensures the minting wallet has to pay a configured amount of SOL to a configured destination wallet.
- Start Date: This guard ensures minting can only start after the configured time.
- Mint Limit: This guard ensures each wallet cannot mint more than a configured amount.

Sugar

Sugar is a CLI to be used with metaplex, and a replacement for the javascript CLI.

INSTALLATION

Mac / Linux / WSL

```
bash <(curl -sSf https://sugar.metaplex.com/install.sh)
```

```
✅ Installation successful: type 'sugar' to start using it.
gitpod /workspace/SolanaBootcampOctober (main) $ sugar
sugar-cli 1.1.0
Command line tool for creating and managing Metaplex Candy Machines.

USAGE:
    sugar [OPTIONS] <SUBCOMMAND>

OPTIONS:
    -h, --help                     Print help information
    -l, --log-level <LOG_LEVEL>    Log level: trace, debug, info, warn, error, off
    -V, --version                  Print version information

SUBCOMMANDS:
    bundlr            Interact with the bundlr network
    collection        Manage the collection on the candy machine
    create-config     Interactive process to create the config file
    deploy            Deploy cache items into candy machine config on-chain
    freeze            Commands for the Candy Machine Freeze feature
    hash              Generate hash of cache file for hidden settings
    help              Print this message or the help of the given subcommand(s)
    launch            Create a candy machine deployment from assets
    mint              Mint one NFT from candy machine
    reveal            Reveal the NFTs from a hidden settings candy machine
    show              Show the on-chain config of an existing candy machine
    sign              Sign one or all NFTs from candy machine
    thaw              Thaw a NFT or all NFTs in a candy machine
    unfreeze-funds    Unlock treasury funds after freeze is turned off or expires
    update            Update the candy machine config on-chain
    upload            Upload assets to storage and creates the cache config
    validate          Validate JSON metadata files
    verify            Verify uploaded data
    withdraw          Withdraw funds from candy machine account closing it
gitpod /workspace/SolanaBootcampOctober (main) $ sugar -V
sugar-cli 1.1.0
gitpod /workspace/SolanaBootcampOctober (main) $ ›
```

Sugar will then use these settings by default if you don't specify them as CLI options, allowing commands to be much simpler.

Prepare Metaplex

Install the solana CLI if not already installed

Generate a new keypair

```
solana-keygen new --outfile ~/.config/solana/devnet.json
```

Set the config

```
solana config set --keypair ~/.config/solana/devnet.json
```

```
solana config set --url https://metaplex.devnet.rpcpool.com/
```

Check the config

```
solana config get
```

Sugar Quickstart

1. Create an assets directory
   `mkdir assets`
2. In this directory add the assets and asset json files. You can find an example in the repo

```
{
    "name": "Studious Crab #1",
    "symbol": "CRAB",
    "description": "The Studious Crabs are smart and productive crabs.",
    "image": "0.png",
    "attributes": [
        {
            "trait_type": "accessory",
            "value": "lamp"
        },
        {
            "trait_type": "chair",
            "value": "red"
        },
        {
            "trait_type": "books",
            "value": "blue"
        }
    ],
    "properties": {
        "files": [
            {
                "uri": "0.png",
                "type": "image/png"
            }
        ]
    }
}
```

From the project directory run

`sugar create-config`

This will take you through a number of choices, you can read more about the choices here
At the end it will save a config file which will look like

```
{
  "price": 1.0,
  "number": 10,
```

```
  "gatekeeper": null,
  "creators": [
    {
      "address": "PanbgtcTiZ2PveV96t2FHSffiLHXXjMuhvoabUUKKm8",
      "share": 100
    }
  ],
  "solTreasuryAccount": "PanbgtcTiZ2PveV96t2FHSffiLHXXjMuhvoabUUKKm8",
  "splTokenAccount": null,
  "splToken": null,
  "goLiveDate": "11 Aug 2022 18:19:16 +0000",
  "endSettings": null,
  "whitelistMintSettings": null,
  "hiddenSettings": null,
  "freezeTime": null,
  "uploadMethod": "bundlr",
  "retainAuthority": true,
  "isMutable": true,
  "symbol": "TEST",
  "sellerFeeBasisPoints": 500,
  "awsS3Bucket": null,
  "nftStorageAuthToken": null,
  "shdwStorageAccount": null
}
```


Once the config is saved you can upload the files

`sugar upload`

You can then create and deploy a candy machine with

`sugar deploy`

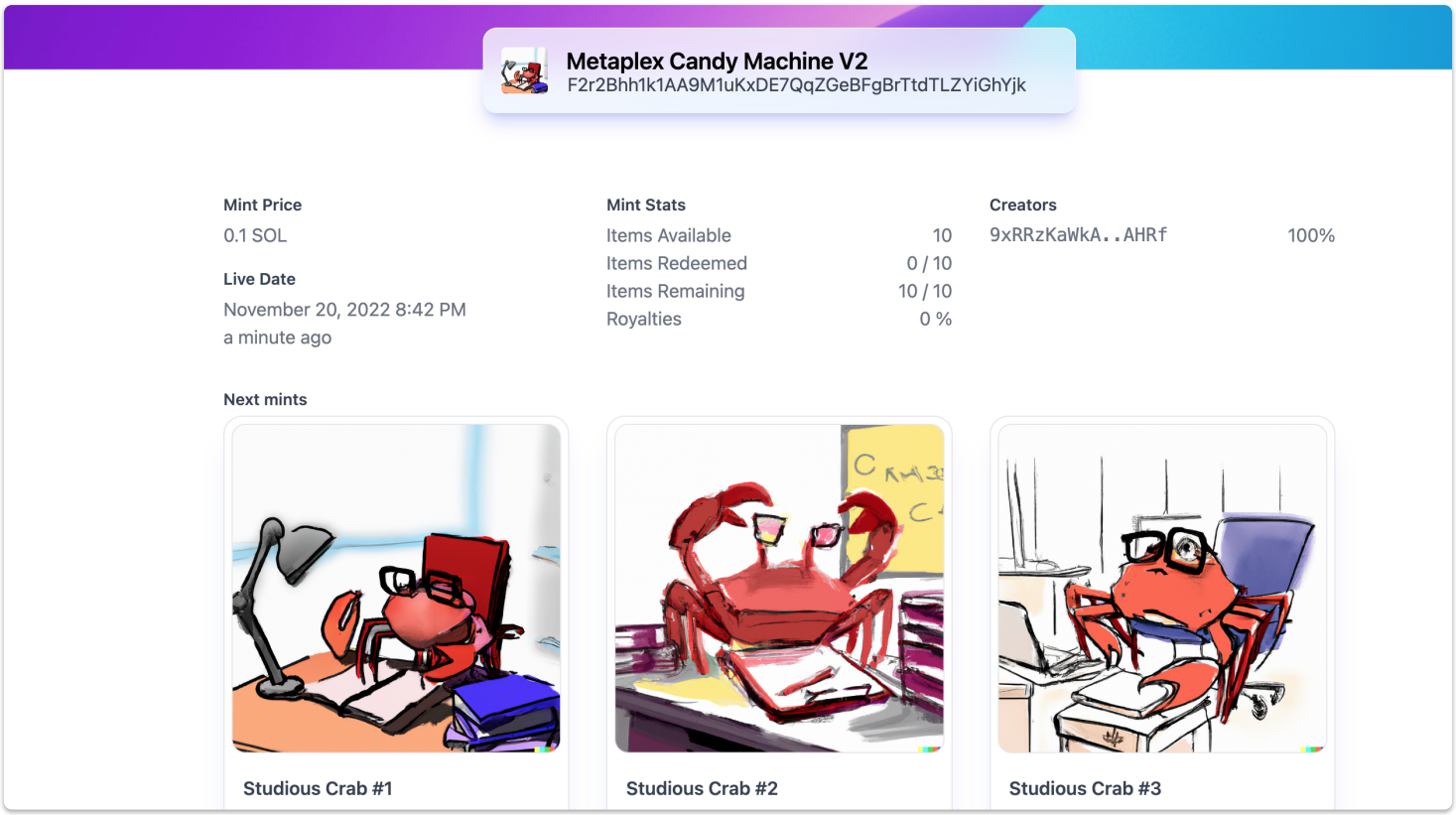
Verify that it has all worked with

`sugar verify`

you can get the details of the candy machine with

`sugar show`

You will get a URL to show your candy machine



Minting NFTs

Now that we have our candy machine deployed we can use it to mint NFTs

`sugar mint`

will mint an NFT to your wallet.

Creating a UI for your Candy Machine

Clone the directory into your project folder

`git clone https://github.com/metaplex-foundation/candy-machine-ui`

Get your Candy Machine ID, either from

`sugar show`

or from the cache.json file

Add the ID to the `.env.example` file and save the file as

`.env`

From within the candy-machine-ui directory run

`yarn install && yarn start`

Candy Machine SDKs

A Javascript SDK is available, Swift and Kotlin SDKs are being planned.
Rust crates are also available

Using the javascript SDK

Setup

You need to setup a connection

```
import { Metaplex } from "@metaplex-foundation/js";
import { Connection, clusterApiUrl } from "@solana/web3.js";

const connection = new Connection(clusterApiUrl("mainnet-beta"));
```

You can then create a Metaplex instance

```
const wallet = Keypair.generate();

const metaplex = Metaplex.make(connection)
    .use(keypairIdentity(wallet))
    .use(bundlrStorage());
```

The metaplex instance then gives the nft module

```
const nftClient = metaplex.nfts();
```

which has useful methods

- `findByMint`
- `findAllByMintList`
- `load`
- `findAllByOwner`
- `findAllByCreator`
- `uploadMetadata`
- `create`
- `update`
- `printNewEdition`
- `use`

See the full API here

Example for uploading metadata and creating an NFT.
The ownership and authority information would be taken from the user set up earlier.

```
const { uri } = await metaplex
  .nfts()
  .uploadMetadata({
    name: "My off-chain name",
    description: "My off-chain description",
    image: "https://arweave.net/123",
  };

const { nft } = await metaplex
  .nfts()
  .create({
    uri,
    name: 'My on-chain NFT',
    sellerFeeBasisPoints: 250, // 2.5%
  };
```

Other useful programs

Gumdrop

See Docs
This helps with creating token and NFT airdrops.

Other considerations

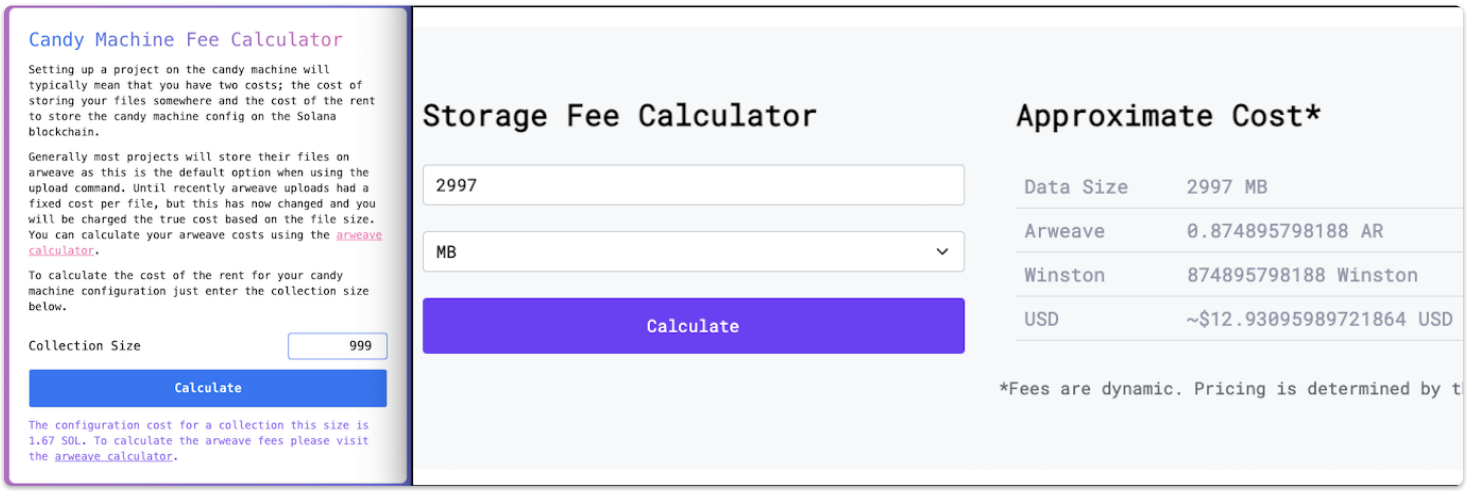
Costs

Solana deployment fees for the collection can be calculated using this online tool
Arweave deployment fees for the collection can be calculated using this tool.

A collection with 999 items and average off-chain data ~3Mb will cost 1.67 SOL and 0.87 AR.

The total price tag for the deployment of this collection at today's Sol and AR valuations would be ~$30, but during the peak price of crypto same collection would cost in excess of $350.

Fair to say to anyone considering launching their own collection now is a good time.



Marketing and brand management

For an NFT launch to be successful it requires a strategy that includes selecting a target audience, brand creation, marketing or potential celebrity endorsement. Even after the launch continuous community management via a social platform such as discord is likely needed to make sure people do not lose interest especially if the creator is to receive royalties from each resale for perpetuity.

Permanence

It is important to state whether NFT is fixed or whether it can be modified. Generally, NFTs are thought of as immutable and owners might object to suddenly realising their content has been changed. This can be enforced with `is_mutable` field being set to `false`.

On the flip side, should NFT be intentionally upgradable, for example, to reflect loalty points or game character attributes it would require `is_mutable` field being set to `true`. As Arweave data is immutable, new off-chain data deployment will be required which will incur some fees as well as updating of the Arweave URI.

Either approach is valid as long as it is communicated to the users.