Juicebox v4 Naming Updates

1. JBTokenStore

- `issueFor(...)` mints a claimable ERC-20 token. This should be disambiguated (e.g. `deployErc20For(...)`).
- All variables should use "token". All natspec descriptions should use "project token".
- Unclaimed -> Internal. Add "internal" to ambiguous names (e.g. `transferFrom(...)` -> `transferInternalTokenFrom(...)`).
- Claimed -> ERC-20. Use ERC-20 in function names and variables (`_preferErc20Tokens`). Add ERC-20 to ambiguous names (e.g. `PROJECT_ALREADY_HAS_TOKEN` -> `PROJECT_ALREADY_HAS_ERC20_TOKEN`).
- Minor nits (`setFor(...)` -> `setTokenFor(...)`).

2. JBFundingCycleStore

- Funding cycle -> ruleset (*alternatives: epoch, cycle, round*). Epoch is used by Ethereum, Cardano, NEAR, and others.
- We must fix the timestamp <-> struct <-> number trilemma (each of these is sometimes referred to as "funding cycle"). If we go with e.g. ruleset, let's use `rulesetTimestamp`, `rulesetData`, and `rulesetId`/`rulesetNumber`. In natspec: "Ruleset timestamp", "Ruleset data", and "Ruleset ID"/"Ruleset number". ALWAYS specify.
- Reconfigure -> Edit ruleset (e.g. `reconfigureFundingCyclesOf(...)` -> `editRulesetOf(...)`).
- Discount rate -> decay rate? Issuance reduction proved clunky. Open to other ideas here.
- Weight -> Issuance rate.
- The distinction between user properties and intrinsic properties might be confusing. How about intrinsic properties -> internally derived properties, user properties -> user provided properties.
- Init -> setLatestRulesetFor?
- Ballot -> approver.
- Base, eligible, standby, queued, and current cycle feel confusing. How about: base -> lastApproved, standby -> upcomingIfApproved, and eligible -> currentIfApproved, and queued -> upcoming (maybe currentlyUpcoming). Let me know if I'm misinterpreting the functionality here.
- Mock -> simulated? Not too important.

3. JBProjects

No changes needed.

4. JBSplitsStore

- `set(...)` -> `setSplitsFor(...)`.

- `_includesLocked(...)` -> `_includesLockedSplits(...)`.
- Maybe rename contract to `JBSplitStore`.

5. JBPrices

- `priceFor(...)` could be made clearer (like `currencyPriceForBase(...)`) but this might not be worth the change.
- `addFeedFor(...)` -> `addPriceFeedFor(...)`

6. JBOperatorStore

- Rename contract to `JBPermissionStore`.
- Operator can be confusing. Might be worth renaming to "admin" (or "delegate", but this term is already overloaded).
- `setOperator(...)` -> `setPermissionsForOperator(...)` or `setPermissionsForAdmin(...)`.
- `setOperators(...)` -> `setPermissionsForOperators(...)` or `setPermissionsForAdmins(...)`.
- Also consider renaming `JBOperatable` -> e.g. `JBPermissionModifiers`.

7. JBDirectory

Maybe rename directory -> registry (and rename the contract to `JBRegistry` or `JBContractRegistry`). Feels slightly clearer.

8. JBController

- Possibly controller -> manager or coordinator. It doesn't really "control" anything, it coordinates other contracts and manages the project lifecycle. That being said, controller is close enough and I would understand keeping it.
- `reservedTokenBalanceOf` -> `undistributedReservedTokenBalanceOf`
- Incorporate changes from `JBFundingCycleStore`
- `totalOutstandingTokensOf(...)` feels confusing. The token store's `totalSupplyOf(...)` feels more like the "outstanding tokens", and this feels like the total supply. Will think about this some more.
- `launchProjectFor(...)` -> `createAndSetupProjectFor(...)`.
- `launchFundingCyclesFor(...)` -> `setupFundingCycleFor(...)` (`setupRulesetFor(...)`). Or create ruleset.
- Maybe `mintTokensOf(...)` -> `mintTokens(...)` (and `burnTokens(...)`) since these functions specify beneficiaries. Meh on this.
- `_configure` -> e.g. `_configureRulesetOf(...)`
- Remove overflow allowance.
- `migrate(...)` -> `migrateController(...)`.

9. JBFundAccessConstraintsStore

- `setFor(...)` -> `setFundAccessConstraintsFor(...)`.
- Distribution limit -> Payout limit or max payout. Cycle payout limit feels good too.
- Rename to `JBFundAccessConstraintStore`.

10. Payment Terminals

Contract naming:

- Payment terminal -> payment processor. Consequently,
- `IJBPaymentTerminal` -> `IJBPayProcessor`
- `IJBRedemptionTerminal` -> `IJBRedeemProcessor`
- `IJBPayoutTerminal` -> `IJBPayoutProcessor`
- `IJBAllowanceTerminal` will be removed.
- `IJBSingleTokenPaymentTerminal` -> `IJBSingleTokenPayProcessor`
- `IJBPayoutRedemptionPaymentTerminal` -> `IJBPayRedeemPayoutProcessor`
- `JBETHPaymentTerminal` -> `JBNativeFundProcessor`. The current name doesn't sufficiently distinguish from `IJBPaymentTerminal` (pay only) despite implementing `IJBPayoutRedemptionPaymentTerminal`.
- `JBERC20PaymentTerminal` -> `JBERC20FundProcessor`.

Other:

- `currentEthOverflowOf` -> `excessNativeTokensOf` OR `redeemableNativeTokenAmountOf`. I prefer `excessNativeTokensOf`.
- `migrate(...)` -> `migrateTerminal(...)`
- `_baseWeightCurrency` -> `_pricingCurrency`? Meh.
- Refund held fees -> unlock held fees.
- Process fees -> collect held fees.

11. JBToken

Rename contract to `JBERC20Token`.

12. JBChainlinkV3PriceFeed

No changes needed.

13. JBETHERC20ProjectPayer/JBETHERC20ProjectPayerDeployer

- Project payer feels vague, but the alternatives seem worse. Considering pay relay, fund forwarder, payment router.
- `setDefaultValues(...)` -> `setDefaultPayParameters(...)`.

14. JBETHERC20SplitsPayer/JBETHERC20SplitsPayerDeployer

Same as project payer.

15. JBMigrationOperator

Will (presumably) be removed.

16. JBReconfigurationBufferBallot

- Ballot -> approver (or e.g. ruleset approver).
- State -> Approval status. If this is too much, state is also good.
- Reconfiguration buffer ballot -> Deadline approver?

17. Other

- Pay/redeem data source -> pay/redeem data provider. Still thinking about this one.
- Pay/redeem delegate -> pay/redeem callback.