cardano-cli

Legenda

< > = Parentesi acute. Definiscono il tipo di informazione da fornire.
( ) = Parentesi tonde. Rappresentano opzioni obbligatorie.
[ ] = Parentesi quadre. Rappresentano opzioni non obbligatorie.
| = Un separatore che indica la scelta da fare tra diverse opzioni racchiuse tra le varie parentesi.

Cardano-cli:~$ cheat sheet V8.0.0 (Italiano)

Questo diagramma è una versione stampabile con un formato 11x17 di 5 pagine per la versione 8.0.0 di cardano-cli

cardano-cli address

cardano-cli address key-gen
```
[--normal-key | --extended-key | --byron-key]
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli address key-hash
```
( --payment-verification-key <STRING>
| --payment-verification-key-file <FILE>
)
[--out-file <FILE>]
```

cardano-cli address build
```
( --payment-verification-key <STRING>
| --payment-verification-key-file <FILE>
| --payment-script-file <FILE>
)
[ --stake-verification-key <STRING>
| --stake-verification-key-file <FILE>
| --stake-script-file <FILE>
|--stake-address <ADDRESS>
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli address info
```
--address <ADDRESS>
[--out-file <FILE>]
```

cardano-cli stake-address

cardano-cli stake-address key-gen
```
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli stake-address build
```
( --stake-verification-key <STRING>
| --stake-verification-key-file <FILE>
| --stake-script-file <FILE>
)
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli stake-address key-hash
```
( --stake-verification-key <STRING>
| --stake-verification-key-file <FILE>
)
[--out-file <FILE>]
```

cardano-cli stake-address registration-certificate
```
( --stake-verification-key <STRING>
| --stake-verification-key-file <FILE>
| --stake-script-file <FILE>
|--stake-address <ADDRESS>
)
--out-file <FILE>
```

cardano-cli stake-address deregistration-certificate
```
( --stake-verification-key <STRING>
| --stake-verification-key-file <FILE>
| --stake-script-file <FILE>
|--stake-address <ADDRESS>
)
--out-file <FILE>
```

cardano-cli stake-address delegation-certificate
```
( --stake-verification-key <STRING>
| --stake-verification-key-file <FILE>
| --stake-script-file <FILE>
|--stake-address <ADDRESS>
)
( --stake-pool-verification-key <STRING>
| --cold-verification-key-file <FILE>
| --stake-pool-id <STAKE-POOL-ID>
)
--out-file <FILE>
```

cardano-cli key

cardano-cli key verification-key
```
--signing-key-file <FILE>
--verification-key-file <FILE>
```

cardano-cli key non-extended-key
```
--extended-verification-key-file <FILE>
--verification-key-file <FILE>
```

cardano-cli key convert-byron-key
```
[--password <TEXT>]
( --byron-payment-key-type
| --legacy-byron-payment-key-type
| --byron-genesis-key-type
| --legacy-byron-genesis-key-type
| --byron-genesis-delegate-key-type
| --legacy-byron-genesis-delegate-key-type
)
(--byron-signing-key-file <FILE> | --byron-verification-key-file <FILE>)
--out-file <FILE>
```

cardano-cli key convert-byron-genesis-vkey
```
--byron-genesis-verification-key <BASE64>
--out-file <FILE>
```

cardano-cli key convert-itn-key
```
(--itn-signing-key-file <FILE> | --itn-verification-key-file <FILE>)
--out-file <FILE>
```

cardano-cli key convert-itn-extended-key
```
--itn-signing-key-file <FILE>
--out-file <FILE>
```

cardano-cli key convert-itn-bip32-key
```
--itn-signing-key-file <FILE>
--out-file <FILE>
```

cardano-cli key convert-cardano-address-key
```
( --shelley-payment-key
| --shelley-stake-key
| --icarus-payment-key
| --byron-payment-key
)
--signing-key-file <FILE>
--out-file <FILE>
```

cardano-cli stake-pool

cardano-cli stake-pool registration-certificate
```
( --stake-pool-verification-key <STRING>
| --cold-verification-key-file <FILE>
)
(--vrf-verification-key <STRING> | --vrf-verification-key-file <FILE>)
--pool-pledge <LOVELACE>
--pool-cost <LOVELACE>
--pool-margin <RATIONAL>
( --pool-reward-account-verification-key <STRING>
| --pool-reward-account-verification-key-file <FILE>
)
( --pool-owner-verification-key <STRING>
| --pool-owner-stake-verification-key-file <FILE>
)
[ [--pool-relay-ipv4 <STRING>
  [--pool-relay-ipv6 <STRING>]
  --pool-relay-port <INT>
| --single-host-pool-relay <STRING> [--pool-relay-port <INT>]
| --multi-host-pool-relay <STRING>
]
[--metadata-url <URL> --metadata-hash <HASH>]
(--mainnet | --testnet-magic <NATURAL>)
--out-file <FILE>
```

cardano-cli stake-pool deregistration-certificate
```
( --stake-pool-verification-key <STRING>
| --cold-verification-key-file <FILE>
)
--epoch <NATURAL>
--out-file <FILE>
```

cardano-cli stake-pool id
```
( --stake-pool-verification-key <STRING>
| --cold-verification-key-file <FILE>
)
[--output-format <STRING>]
```

cardano-cli stake-pool metadata-hash
```
--pool-metadata-file <FILE>
[--out-file <FILE>]
```

cardano-cli query

cardano-cli query protocol-parameters
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query tip
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query stake-pools
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query stake-distribution
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query stake-address-info
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
--address <ADDRESS>
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query utxo
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--whole-utxo | (--address <ADDRESS>) | (--tx-in <TX-IN>))
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query ledger-state
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query protocol-state
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--out-file <FILE>]
```

cardano-cli query stake-snapshot
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[ --all-stake-pools | [--stake-pool-id <STAKE-POOL-ID>]]
[--out-file <FILE>]
```

cardano-cli query pool-state
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
--stake-pool-id <STAKE-POOL-ID>
```

cardano-cli query leadership-schedule
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
--genesis <FILE>
( --stake-pool-verification-key <STRING>
| --cold-verification-key-file <FILE>
| --stake-pool-id <STAKE-POOL-ID>
)
--vrf-signing-key-file <FILE>
(--current | --next)
[--out-file <FILE>]
```

cardano-cli query kes-period-info
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
--op-cert-file <FILE>
[--out-file <FILE>]
```

cardano-cli query tx-mempool
```
--socket-path <SOCKET_PATH>
[ --shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
(info | next-tx | tx-exists)
[--out-file <FILE>]
```
Nuovo Comando 8.0.0

---

cardano-cli genesis

cardano-cli genesis key-gen-genesis
```
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli genesis key-gen-delegate
```
--verification-key-file <FILE>
--signing-key-file <FILE>
--operational-certificate-issue-counter-file <FILE>
```

cardano-cli genesis key-gen-utxo
```
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli genesis key-hash
```
--verification-key-file <FILE>
```

cardano-cli genesis get-ver-key
```
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli genesis initial-addr
```
--verification-key-file FILE
(--mainnet | --testnet-magic NATURAL)
[--out-file FILE]
```

cardano-cli genesis initial-txin
```
--verification-key-file FILE
(--mainnet | --testnet-magic NATURAL)
[--out-file FILE]
```

cardano-cli genesis create-cardano
```
--genesis-dir <DIR>
[--gen-genesis-keys <INT>]
[--gen-utxo-keys <INT>]
[--start-time <UTC-TIME>]
[--supply <LOVELACE>]
[--security-param <INT>]
[--slot-length <INT>]
[--slot-coefficient <RATIONAL>]
(--mainnet | --testnet-magic <NATURAL>)
--byron-template <FILEPATH>
--shelley-template <FILEPATH>
--alonzo-template <FILEPATH>
[--node-config-template <FILEPATH>]
```

cardano-cli genesis create
```
--genesis-dir <DIR>
[--gen-genesis-keys <INT>]
[--gen-utxo-keys <INT>]
[--start-time <UTC-TIME>]
[--supply <LOVELACE>]
(--mainnet | --testnet-magic <NATURAL>)
```

cardano-cli genesis create-staked
```
--genesis-dir <DIR>
[--gen-genesis-keys <INT>]
[--gen-utxo-keys <INT>]
[--gen-pools <INT>]
[--gen-stake-delegs <INT>]
[--start-time <UTC-TIME>]
[--supply <LOVELACE>]
[--supply-delegated <LOVELACE>]
(--mainnet | --testnet-magic <NATURAL>)
[--bulk-pool-cred-files <INT>]
[--bulk-pools-per-file <INT>]
[--num-stuffed-utxo <INT>]
[--relay-specification-file <FILE>]
```

cardano-cli genesis hash
```
--genesis <FILE>
```

---

cardano-cli text-view

cardano-cli text-view decode-cbor
```
--in-file <FILE>
[--out-file <FILE>]
```

cardano-cli governance

cardano-cli governance create-update-proposal

```
--out-file <FILE>
--epoch <NATURAL>
(--genesis-verification-key-file <FILE>)
[--protocol-major-version <NATURAL> --protocol-minor-version <NATURAL>]
[--decentralization-parameter <RATIONAL>]
[--extra-entropy <HEX> | --reset-extra-entropy]
[--max-block-header-size <NATURAL>]
[--max-block-body-size <NATURAL>]
[--max-tx-size <NATURAL>]
[--min-fee-constant <LOVELACE>]
[--min-fee-linear <NATURAL>]
[--min-utxo-value <NATURAL>]
[--key-reg-deposit-amt <NATURAL>]
[--pool-reg-deposit <NATURAL>]
[--min-pool-cost <NATURAL>]
[--pool-retirement-epoch-boundary <INT>]
[--number-of-pools <NATURAL>]
[--pool-influence <RATIONAL>]
[--monetary-expansion <RATIONAL>]
[--treasury-expansion <RATIONAL>]
[--utxo-cost-per-word <LOVELACE>]
[--price-execution-steps <RATIONAL> --price-execution-memory <RATIONAL>]
[--max-tx-execution-units (<INT, INT>)]
[--max-block-execution-units (<INT, INT>)]
[--max-value-size <INT>]
[--collateral-percent <INT>]
[--max-collateral-inputs <INT>]
[--utxo-cost-per-byte <LOVELACE>]
[--cost-model-file <FILE>]
```

cardano-cli governance create-genesis-key-delegation-certificate

```
( --genesis-verification-key <STRING>
| --genesis-verification-key-file <FILE>
| --genesis-verification-key-hash <STRING>
)
( --genesis-delegate-verification-key <STRING>
| --genesis-delegate-verification-key-file <FILE>
| --genesis-delegate-verification-key-hash <STRING>
)
(--vrf-verification-key <STRING>
| --vrf-verification-key-file <FILE>
| --vrf-verification-key-hash <STRING>
)
--out-file <FILE>
```

cardano-cli governance create-poll — Nuovo Comando

```
--question <STRING>
(--answer <STRING>)
[--nonce <UINT>]
--out-file <FILE>
```

cardano-cli governance answer-poll — Nuovo Comando

```
--poll-file <FILE>
[--answer <INT>]
[--out-file <FILE>]
```

cardano-cli governance verify-poll — Nuovo Comando

```
--poll-file <FILE>
--tx-file <FILE>
[--out-file <FILE>]
```

cardano-cli governance create-mir-certificate

```
( (--reserves | --treasury)
(--stake-address ADDRESS)
(--reward LOVELACE)
--out-file FILE
| stake-addresses
| transfer-to-treasury
| transfer-to-rewards
)
```

cardano-cli governance create-mir-certificate stake-addresses

```
(--reserves | --treasury)
(--stake-address ADDRESS)
(--reward LOVELACE)
--out-file FILE
```

cardano-cli governance create-mir-certificate transfer-to-treasury

```
--transfer <LOVELACE>
--out-file <FILE>
```

cardano-cli governance create-mir-certificate transfer-to-rewards

```
--transfer <LOVELACE>
--out-file <FILE>
```

cardano-cli node

cardano-cli node key-gen

```
--cold-verification-key-file <FILE>
--cold-signing-key-file <FILE>
--operational-certificate-issue-counter-file <FILE>
```

cardano-cli node key-gen-KES

```
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli node key-gen-VRF

```
--verification-key-file <FILE>
--signing-key-file <FILE>
```

cardano-cli node key-hash-VRF

```
(--verification-key <STRING> | --verification-key-file <FILE>)
[--out-file <FILE>]
```

cardano-cli node new-counter

```
(--stake-pool-verification-key <STRING>
| --genesis-delegate-verification-key <STRING>
| --cold-verification-key-file <FILE>
)
--counter-value <INT>
--operational-certificate-issue-counter-file <FILE>
```

cardano-cli node issue-op-cert

```
(--kes-verification-key <STRING> | --kes-verification-key-file <FILE>)
--cold-signing-key-file <FILE>
--operational-certificate-issue-counter-file <FILE>
--kes-period <NATURAL>
--out-file <FILE>
```

cardano-cli transaction

cardano-cli transaction sign

```
(--tx-body-file <FILE> | --tx-file <FILE>)
(--signing-key-file <FILE> [--address <STRING>])
[--mainnet | --testnet-magic <NATURAL>]
--out-file <FILE0
```

cardano-cli transaction witness

```
--tx-body-file <FILE>
--signing-key-file <FILE>
[--address <STRING>]
[--mainnet | --testnet-magic <NATURAL>]
--out-file <FILE>
```

cardano-cli transaction assemble

```
--tx-body-file <FILE>
(--witness-file <FILE>)
--out-file <FILE>
```

cardano-cli transaction submit

```
[--socket-path <SOCKET_PATH>]
[--shelley-mode
| --byron-mode [--epoch-slots <NATURAL>]
| --cardano-mode [--epoch-slots <NATURAL>]
]
(--mainnet | --testnet-magic <NATURAL>)
--tx-file <FILE>
```

cardano-cli transaction policyid

```
--script-file <FILE>
```

cardano-cli transaction calculate-min-fee

```
--tx-body-file <FILE>
[--mainnet | --testnet-magic <NATURAL>]
--protocol-params-file <FILE>
--tx-in-count <NATURAL>
--tx-out-count <NATURAL>
--witness-count <NATURAL>
[--byron-witness-count <NATURAL>]
```

cardano-cli transaction txid

```
(--tx-body-file <FILE> | --tx-file <FILE>)
```

cardano-cli transaction view

```
(--tx-body-file <FILE> | --tx-file <FILE>)
```

cardano-cli transaction hash-script-data

```
( --script-data-cbor-file <CBOR FILE>
| --script-data-file <JSON FILE>
| --script-data-value <JSON VALUE>
)
```

cardano-cli transaction calculate-min-required-utxo

```
[ --byron-era
| --shelley-era
| --allegra-era
| --mary-era
| --alonzo-era
| --babbage-era
]
--protocol-params-file <FILE>
--tx-out <ADDRESS VALUE>
[ --tx-out-datum-hash <HASH>
| --tx-out-datum-hash-cbor-file <CBOR FILE>
| --tx-out-datum-hash-file <JSON FILE>
| --tx-out-datum-hash-value <JSON VALUE>
| --tx-out-datum-embed-cbor-file <CBOR FILE>
| --tx-out-datum-embed-file <JSON FILE>
| --tx-out-datum-embed-value <JSON VALUE>
| --tx-out-inline-datum-cbor-file <CBOR FILE>
| --tx-out-inline-datum-file <JSON FILE>
| --tx-out-inline-datum-value <JSON VALUE>
]
[--tx-out-reference-script-file <FILE>]
```

cardano-cli transaction build-raw

```
[ --byron-era
| --shelley-era
| --allegra-era
| --mary-era
| --alonzo-era
| --babbage-era
]
[--script-valid | --script-invalid]
(--tx-in <TX-IN>
  [ --spending-tx-in-reference <TX-IN>
    --spending-plutus-script-v2
    ( --spending-reference-tx-in-datum-cbor-file <CBOR FILE>
    | --spending-reference-tx-in-datum-file <JSON FILE>
    | --spending-reference-tx-in-datum-value <JSON VALUE>
    | --spending-reference-tx-in-inline-datum-present
    )
    ( --spending-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --spending-reference-tx-in-redeemer-file <JSON FILE>
    | --spending-reference-tx-in-redeemer-value <JSON VALUE>
    )
    --spending-reference-tx-in-execution-units <(INT, INT)>
  | --simple-script-tx-in-reference <TX-IN>
  | --tx-in-script-file <FILE>
    [
      ( --tx-in-datum-cbor-file <CBOR FILE>
      | --tx-in-datum-file <JSON FILE>
      | --tx-in-datum-value <JSON VALUE>
      | --tx-in-inline-datum-present
      )
      ( --tx-in-redeemer-cbor-file <CBOR FILE>
      | --tx-in-redeemer-file <JSON FILE>
      | --tx-in-redeemer-value <JSON VALUE>
      )
      --tx-in-execution-units <(INT, INT)>]
  ])
[--read-only-tx-in-reference <TX-IN>]
[--tx-in-collateral <TX-IN>]
[--tx-out-return-collateral <ADDRESS VALUE>]
[--tx-total-collateral <INTEGER>]
[--required-signer <FILE> | --required-signer-hash <HASH>]
[--tx-out <ADDRESS VALUE>
  [ --tx-out-datum-hash <HASH>
  | --tx-out-datum-hash-cbor-file <CBOR FILE>
  | --tx-out-datum-hash-file <JSON FILE>
  | --tx-out-datum-hash-value <JSON VALUE>
  | --tx-out-datum-embed-cbor-file <CBOR FILE>
  | --tx-out-datum-embed-file <JSON FILE>
  | --tx-out-datum-embed-value <JSON VALUE>
  | --tx-out-inline-datum-cbor-file <CBOR FILE>
  | --tx-out-inline-datum-file <JSON FILE>
  | --tx-out-inline-datum-value <JSON VALUE>
  ]
  [--tx-out-reference-script-file <FILE>]]
[--mint <VALUE>
  ( --mint-script-file <FILE>
    [
      ( --mint-redeemer-cbor-file <CBOR FILE>
      | --mint-redeemer-file <JSON FILE>
      | --mint-redeemer-value <JSON VALUE>
      )
      --mint-execution-units <(INT, INT)>]
  | --simple-minting-script-tx-in-reference <TX-IN> --policy-id <HASH>
  | --mint-tx-in-reference <TX-IN>
    --mint-plutus-script-v2
    ( --mint-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --mint-reference-tx-in-redeemer-file <JSON FILE>
    | --mint-reference-tx-in-redeemer-value <JSON VALUE>
    )
    --mint-reference-tx-in-execution-units <(INT, INT)>
    --policy-id <HASH>
  )]
[--invalid-before <SLOT>]
[--invalid-hereafter <SLOT>]
[--fee <LOVELACE>]
[--certificate-file <CERTIFICATEFILE>
  [ --certificate-script-file <FILE>
    [
      ( --certificate-redeemer-cbor-file <CBOR FILE>
      | --certificate-redeemer-file <JSON FILE>
      | --certificate-redeemer-value <JSON VALUE>
      )
      --certificate-execution-units <(INT, INT)>]
  | --certificate-tx-in-reference <TX-IN>
    --certificate-plutus-script-v2
    ( --certificate-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --certificate-reference-tx-in-redeemer-file <JSON FILE>
    | --certificate-reference-tx-in-redeemer-value <JSON VALUE>
    )
    --certificate-reference-tx-in-execution-units <(INT, INT)>
  ]]
[--withdrawal <WITHDRAWAL>
  [ --withdrawal-script-file <FILE>
    [
      ( --withdrawal-redeemer-cbor-file <CBOR FILE>
      | --withdrawal-redeemer-file <JSON FILE>
      | --withdrawal-redeemer-value <JSON VALUE>
      )
      --withdrawal-execution-units <(INT, INT)>]
  | --withdrawal-tx-in-reference <TX-IN>
    --withdrawal-plutus-script-v2
    ( --withdrawal-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --withdrawal-reference-tx-in-redeemer-file <JSON FILE>
    | --withdrawal-reference-tx-in-redeemer-value <JSON VALUE>
    )
    --withdrawal-reference-tx-in-execution-units <(INT, INT)>
  ]]
[--json-metadata-no-schema | --json-metadata-detailed-schema]
[--auxiliary-script-file <FILE>]
[--metadata-json-file <FILE> | --metadata-cbor-file <FILE>]
[--genesis <FILE> | --protocol-params-file <FILE>]
[--update-proposal-file <FILE>]
--out-file <FILE>
```

cardano-cli transaction build

```
[--socket-path <SOCKET_PATH>]
[ --byron-era
| --shelley-era
| --allegra-era
| --mary-era
| --alonzo-era
| --babbage-era
]
[--shelley-mode
| --byron-mode [--epoch-slots <SLOTS>]
| --cardano-mode [--epoch-slots <SLOTS>]
]
(--mainnet | --testnet-magic <NATURAL>)
[--script-valid | --script-invalid]
[--witness-override <WORD>]
(--tx-in <TX-IN>
  [ --spending-tx-in-reference <TX-IN>
    --spending-plutus-script-v2
    ( --spending-reference-tx-in-datum-cbor-file <CBOR FILE>
    | --spending-reference-tx-in-datum-file <JSON FILE>
    | --spending-reference-tx-in-datum-value <JSON VALUE>
    | --spending-reference-tx-in-inline-datum-present
    )
    ( --spending-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --spending-reference-tx-in-redeemer-file <JSON FILE>
    | --spending-reference-tx-in-redeemer-value <JSON VALUE>
    )
  | --simple-script-tx-in-reference <TX-IN>
  | --tx-in-script-file <FILE>
    [
      ( --tx-in-datum-cbor-file <CBOR FILE>
      | --tx-in-datum-file <JSON FILE>
      | --tx-in-datum-value <JSON VALUE>
      | --tx-in-inline-datum-present
      )
      ( --tx-in-redeemer-cbor-file <CBOR FILE>
      | --tx-in-redeemer-file <JSON FILE>
      | --tx-in-redeemer-value <JSON VALUE>
      )
  ])
[--read-only-tx-in-reference <TX-IN>]
[--required-signer <FILE> | --required-signer-hash <HASH>]
[--tx-in-collateral <TX-IN>]
[--tx-out-return-collateral <ADDRESS VALUE>]
[--tx-total-collateral <INTEGER>]
[--tx-out <ADDRESS VALUE>
  [ --tx-out-datum-hash <HASH>
  | --tx-out-datum-hash-cbor-file <CBOR FILE>
  | --tx-out-datum-hash-file <JSON FILE>
  | --tx-out-datum-hash-value <JSON VALUE>
  | --tx-out-datum-embed-cbor-file <CBOR FILE>
  | --tx-out-datum-embed-file <JSON FILE>
  | --tx-out-datum-embed-value <JSON VALUE>
  | --tx-out-inline-datum-cbor-file <CBOR FILE>
  | --tx-out-inline-datum-file <JSON FILE>
  | --tx-out-inline-datum-value <JSON VALUE>
  ]
  [--tx-out-reference-script-file <FILE>]]
--change-address <ADDRESS>
[--mint <VALUE>
  ( --mint-script-file <FILE>
    [
      ( --mint-redeemer-cbor-file <CBOR FILE>
      | --mint-redeemer-file <JSON FILE>
      | --mint-redeemer-value <JSON VALUE>
      ]
  | --simple-minting-script-tx-in-reference <TX-IN> --policy-id <HASH>
  | --mint-tx-in-reference <TX-IN>
    --mint-plutus-script-v2
    ( --mint-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --mint-reference-tx-in-redeemer-file <JSON FILE>
    | --mint-reference-tx-in-redeemer-value <JSON VALUE>
    )
    --policy-id <HASH>
  )]
[--invalid-before <SLOT>]
[--invalid-hereafter <SLOT>]
[--certificate-file <CERTIFICATEFILE>
  [ --certificate-script-file <FILE>
    [ --certificate-redeemer-cbor-file <CBOR FILE>
    | --certificate-redeemer-file <JSON FILE>
    | --certificate-redeemer-value <JSON VALUE>
    ]
  | --certificate-tx-in-reference <TX-IN>
    --certificate-plutus-script-v2
    ( --certificate-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --certificate-reference-tx-in-redeemer-file <JSON FILE>
    | --certificate-reference-tx-in-redeemer-value <JSON VALUE>
    )
  ]]
[--withdrawal <WITHDRAWAL>
  [ --withdrawal-script-file <FILE>
    [ --withdrawal-redeemer-cbor-file <CBOR FILE>
    | --withdrawal-redeemer-file <JSON FILE>
    | --withdrawal-redeemer-value <JSON VALUE>
    ]
  | --withdrawal-tx-in-reference <TX-IN>
    --withdrawal-plutus-script-v2
    ( --withdrawal-reference-tx-in-redeemer-cbor-file <CBOR FILE>
    | --withdrawal-reference-tx-in-redeemer-file <JSON FILE>
    | --withdrawal-reference-tx-in-redeemer-value <JSON VALUE>
    )
  ]]
[--json-metadata-no-schema | --json-metadata-detailed-schema]
[--auxiliary-script-file <FILE>]
[--metadata-json-file <FILE> | --metadata-cbor-file <FILE>]
[--genesis <FILE> | --protocol-params-file <FILE>]
[--update-proposal-file <FILE>]
(--out-file <FILE> | --calculate-plutus-script-cost <FILE>)
```

cardano-cli ping

Nuovo Comando 8.0.0

```
[-c|--count <COUNT>]
((-h|--host <HOST>) | (-u|--unixsock <SOCKET>))
[-p|--port <PORT>]
[-m|--magic <MAGIC>]
[-j|--json]
[-q|--quiet]
```

```
cardano-cli byron
```

cardano-cli byron key

cardano-cli byron key keygen
```
--secret <FILEPATH>
```

cardano-cli byron key to-verification
```
[--byron-legacy-formats | --byron-formats]
--secret <FILEPATH>
--to <FILEPATH>
```

cardano-cli byron key signing-key-public
```
[--byron-legacy-formats | --byron-formats]
--secret <FILEPATH>
```

cardano-cli byron key signing-key-address
```
[--byron-legacy-formats | --byron-formats]
(--mainnet | --testnet-magic <NATURAL>)
--secret <FILEPATH>
```

cardano-cli byron key migrate-delegate-key-from
```
--from <FILEPATH>
--to <FILEPATH>
```

cardano-cli byron transaction

cardano-cli byron transaction submit-tx
```
(--mainnet | --testnet-magic <NATURAL>)
--tx <FILEPATH>
```

cardano-cli byron transaction issue-genesis-utxo-expenditure
```
--genesis-json <FILEPATH>
(--mainnet | --testnet-magic <NATURAL>)
[--byron-legacy-formats | --byron-formats]
--tx <FILEPATH>
--wallet-key <FILEPATH>
--rich-addr-from <ADDR>
(--txout '(<"ADDR", LOVELACE>)')
```

cardano-cli byron transaction issue-utxo-expenditure
```
(--mainnet | --testnet-magic <NATURAL>)
[--byron-legacy-formats | --byron-formats]
--tx <FILEPATH>
--wallet-key <FILEPATH>
(--txin (<TXID,INDEX>))
(--txout '(<"ADDR", LOVELACE>)')
```

cardano-cli byron transaction txid
```
--tx <FILEPATH>
```

cardano-cli byron genesis

cardano-cli byron genesis genesis
```
--genesis-output-dir <FILEPATH>
--start-time <POSIXSECONDS>
--protocol-parameters-file <FILEPATH>
--k <INT>
--protocol-magic <INT>
--n-poor-addresses <INT>
--n-delegate-addresses <INT>
--total-balance <INT>
--delegate-share <DOUBLE>
--avvm-entry-count <INT>
--avvm-entry-balance <INT>
[--avvm-balance-factor <DOUBLE>]
[--secret-seed <INT>]
```

cardano-cli byron genesis print-genesis-hash
```
--genesis-json <FILEPATH>
```

cardano-cli byron governance

cardano-cli byron governance create-update-proposal
```
(--mainnet | --testnet-magic <NATURAL>)
--signing-key <FILEPATH>
--protocol-version-major <WORD16>
--protocol-version-minor <WORD16>
--protocol-version-alt <WORD8>
--application-name <STRING>
--software-version-num <WORD32>
--system-tag <STRING>
--installer-hash <HASH>
--filepath <FILEPATH>
[--script-version <WORD16>]
[--slot-duration <NATURAL>]
[--max-block-size <NATURAL>]
[--max-header-size <NATURAL>]
[--max-tx-size <NATURAL>]
[--max-proposal-size <NATURAL>]
[--max-mpc-thd <DOUBLE>]
[--heavy-del-thd <DOUBLE>]
[--update-vote-thd <DOUBLE>]
[--update-proposal-thd <DOUBLE>]
[--time-to-live <WORD64>]
[--softfork-init-thd <DOUBLE>
 --softfork-min-thd <DOUBLE>
 --softfork-thd-dec <DOUBLE>]
[--tx-fee-a-constant <INT> --tx-fee-b-constant <DOUBLE>]
[--unlock-stake-epoch <WORD64>]
```

cardano-cli byron governance create-proposal-vote
```
(--mainnet | --testnet-magic <NATURAL>)
--signing-key <FILEPATH>
--proposal-filepath <FILEPATH>
(--vote-yes | --vote-no)
--output-filepath <FILEPATH>
```

cardano-cli byron governance submit-update-proposal
```
(--mainnet | --testnet-magic <NATURAL>)
--filepath <FILEPATH>
```

cardano-cli byron governance submit-proposal-vote
```
(--mainnet | --testnet-magic <NATURAL>)
--filepath <FILEPATH>
```

cardano-cli byron query

cardano-cli byron query get-tip
```
(--mainnet | --testnet-magic NATURAL)
```

cardano-cli byron miscellaneous

cardano-cli byron miscellaneous validate-cbor
```
[ --byron-block <INT>
| --byron-delegation-certificate
| --byron-tx
| --byron-update-proposal
| --byron-vote
]
--filepath <FILEPATH>
```

cardano-cli byron miscellaneous pretty-print-cbor
```
--filepath <FILEPATH>
```