

# Apache Spark: Zpracování v paměti

**Apache Spark** je open-source engine pro paralelní zpracování dat v reálném čase i v dávkách. Jeho hlavní devizou je schopnost udržovat data v operační paměti (In-memory computing), což dramaticky zrychluje opakované dotazy a algoritmy [strojového učení](#).

## 1. Architektura Apache Spark

Spark využívá architekturu typu **Master-Worker**. Centrálním mozkem je **Driver**, který rozděljuje úlohy mezi jednotlivé **Executory** běžící na pracovních uzlech (Worker Nodes).

- **Driver Program:** Hlavní proces, který sestavuje logický plán výpočtu (tzv. DAG).
- **Cluster Manager:** Správce zdrojů (např. Kubernetes, YARN nebo vestavěný Standalone manager), který přiděluje paměť a procesory.
- **Executors:** Procesy na jednotlivých serverech, které provádějí samotné výpočty a ukládají data v RAM.

## 2. Klíčové koncepty a datové struktury

Spark se v průběhu let vyvíjel od nízkoúrovňových struktur k vysoce optimalizovaným rozhraním:

- **RDD (Resilient Distributed Dataset):** Původní stavební kámen. Jde o kolekci objektů rozdělenou napříč clusterem, která je odolná proti chybám (při pádu uzlu se dokáže sama dopočítat).
- **DataFrame:** Podobně jako v [Pandas](#) nebo SQL, data jsou organizována do pojmenovaných sloupců. Tento formát umožňuje Sparku provádět optimalizace pomocí engine **Catalyst**.
- **Lazy Evaluation:** Spark nezačne počítat hned. Pouze si zapisuje seznam operací (transformací), které má provést. Skutečný výpočet se spustí až ve chvíli, kdy klient vyžádá výsledek (tzv. Action).

## 3. Ekosystém Sparku (Knihovny)

Spark je „švýcarský nůž“ pro data, protože v jednom balíku kombinuje různé typy analýz:

| Komponenta             | Účel                                                              |
|------------------------|-------------------------------------------------------------------|
| <b>Spark SQL</b>       | Práce se strukturovanými daty pomocí standardního SQL dotazování. |
| <b>Spark Streaming</b> | Zpracování dat v reálném čase (např. z <a href="#">Kafky</a> ).   |
| <b>MLlib</b>           | Škálovatelná knihovna pro algoritmy strojového učení.             |
| <b>GraphX</b>          | Analýza grafů a sociálních vazeb (např. hledání komunit).         |

## 4. Proč zvolit Spark místo Hadoopu?

| Vlastnost           | Hadoop MapReduce                | Apache Spark                             |
|---------------------|---------------------------------|------------------------------------------|
| <b>Rychlost</b>     | Pomalejší (častý zápis na disk) | Až 100x rychlejší (In-memory)            |
| <b>Náročnost</b>    | Nižší nároky na RAM             | Vysoké nároky na operační paměť          |
| <b>Flexibilita</b>  | Pouze dávkové zpracování        | Dávky, streaming i interaktivní dotazy   |
| <b>Programování</b> | Komplexní Java kód              | Snadné API v Pythonu, Scale, Javě nebo R |

## 5. Použití v praxi

- **ETL procesy:** Čištění a transformace terabytů surových dat z [Data Lake](#) do databází.
- **Prediktivní analýza:** Trénování modelů na obrovských datasetech, které se nevejdou na jeden počítač.
- **Analýza logů:** Vyhledávání bezpečnostních hrozeb v reálném čase analýzou miliard záznamů ze serverů.

Související články:

- [Big Data a model 5V](#)
- [Apache Kafka: Zdroj dat pro Spark](#)
- [Python: Programování v PySpark](#)

Tagy: *it big-data spark analytics data-science in-memory parallel-computing*

From:  
<http://serviceit.cz/> - **IT ENCYKLOPEDIE**

Permanent link:  
[http://serviceit.cz/doku.php?id=it:sw:apache\\_spark](http://serviceit.cz/doku.php?id=it:sw:apache_spark)

Last update: **2026/01/02 20:01**

