Miroslav Holec
Software & Cloud Architect
Microsoft MVP: Microsoft Azure
MCSD, MCSA, MTA

miroslavholec.cz
@miroslavholec

Zvyšujeme výkonnost aplikací s

(Azure) Redis Cache

2017 Global Azure BOOTCAMP

HAVIT SOFTWARE DEVELOPMENT

odkaz.me/redis

Agenda

Úvod do Redis Cache

Co je Redis Cache

Vlastnosti

Nástroje pro správu

Azure Redis Cache

Klíče

Pipelining

Příklady použití v praxi

Dema

Redis Cache nástroje a práce s CLI

Azure Redis Cache

Output Cache / Session State providers

Redis Cache

REmote DIctionary Server

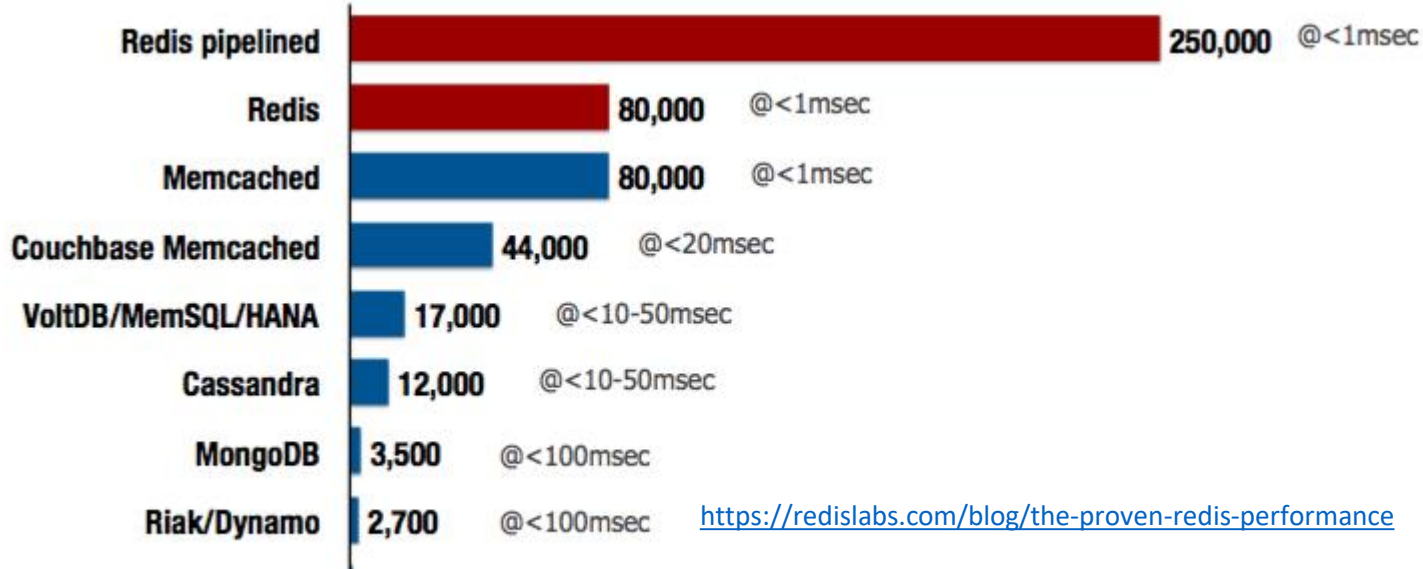
Open source úložiště strukturovaných dat v operační paměti, použitelné jako databáze, cache a message broker. Podporuje datové struktury jako stringy, listy, sety, hashe a další, nad kterými poskytuje speciální operace.

Azure Redis Cache

Služba založená na open-source redis cache poskytující přístup k zabezpečené vyhrazené mezipaměti spravované Microsoftem.

Vlastnosti Redis Cache

- In-memory úložiště -> vysoká výkonnost (propustnost)
- Úložiště typu klíč-hodnota, kde hodnota může být i strukturovaný datový typ
- Single-threaded (atomicita operací)
- Standardně nepersistentní ale podporuje AOF (append-only files), RDB (snapshots)
- Podpora transakcí (izolované, prováděné sekvenčně)
- Logické rozdělení na databáze
- Clustering
- Většina operací řešena s *O*(1)
- Použití jako LRU cache
- Pipelining



https://redislabs.com/blog/the-proven-redis-performance

Nástroje pro správu Redis Cache

```
PM> Install-Package StackExchange.Redis.StrongName
```

https://github.com/StackExchange/StackExchange.Redis

Redis Desktop Manager

http://redisdesktop.com

- GUI pro Win, vhodný pro procházení klíčů

Redsmin

http://redsmin.com

- Online tool, real-time monitoring
- Nástroje pro správu

MSOpenTech/redis

https://github.com/ServiceStack/redis-windows

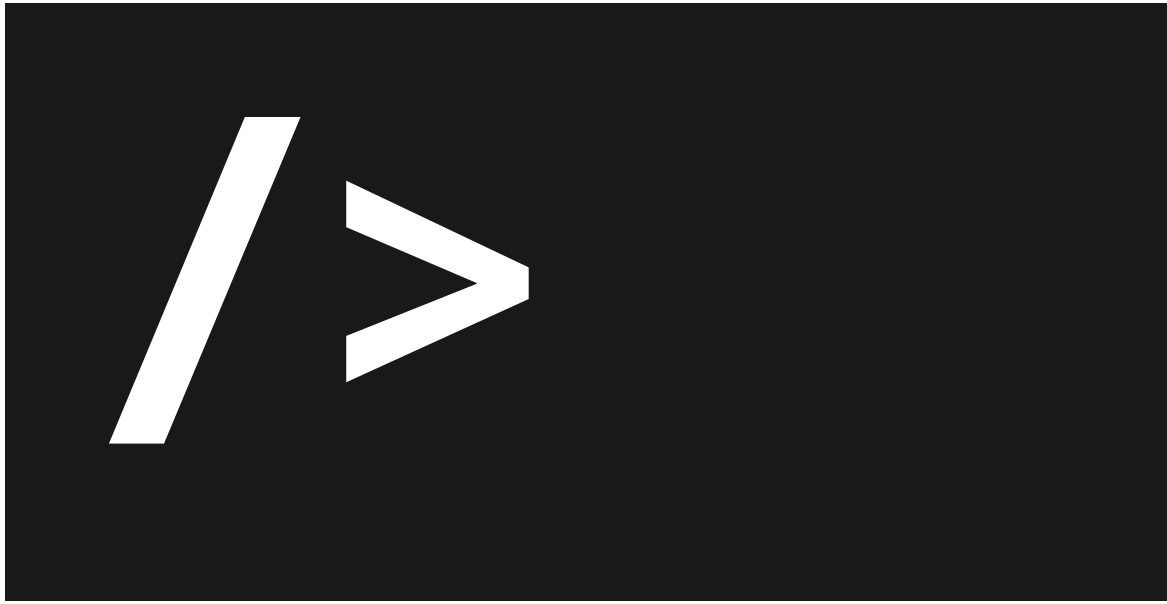
- Port Redis na Win (JEN 64-bit)
- Obsahuje CLI, Redis-Benchmark
- https://github.com/MSOpenTech/redis

Redis Benchmark

- https://redis.io/topics/benchmarks

DEMO

Redis Cache CLI a nástroje



Azure Redis Cache

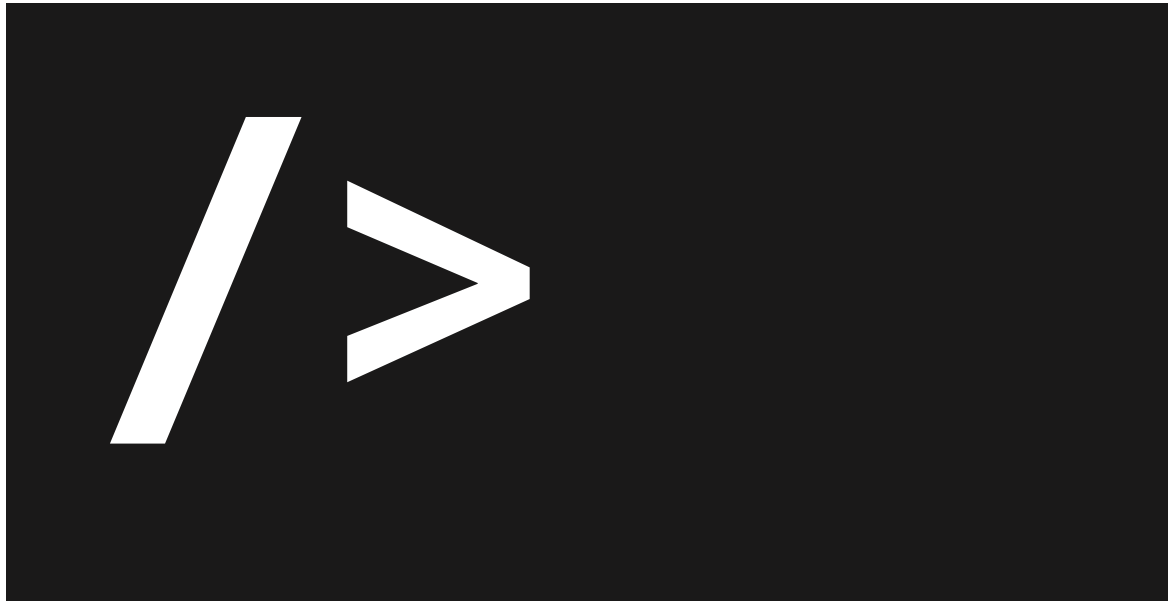
Služba založená na open-source redis cacheposkytující přístup k zabezpečené vyhrazené mezipaměti spravované Microsoftem.

- Microsoft poskytuje servery dle výkonnosti ve 3 plánech (Basic, Standard, Premium)
- Velikost až 53 GB, pomocí clusteru i více - jen premium
- Standard a premium s 99,9% SLA
- Monitoring, diagnostika a alerty
- SSL/non SSL
- Škálování (mizerné z principu)
- RDB Backup, neumí AOF

https://azure.microsoft.com/cs-cz/services/cache

DEMO

Azure Redis Cache



Klíče v Redis Cache

- Klíče jsou stringem o velikosti až 512 MB
- Doporučena kratší délka, logické celky oddělené, například article:1:id
- Životnost klíče lze získat příkazem TTL [key]
- Likvidace klíčů probíhá aktivně i pasivně
- Změna životnosti klíče pomocí EXPIRE [key] [seconds]
- Debugování pomocí DEBUG OBJECT [key]

Všechno o open source Redis Cache: https://redis.io

Použití: Mezipamět - general purpose

IN-MEMORY VS REDIS

LOKÁLNÍ VS DISTRIBUOVANÁ

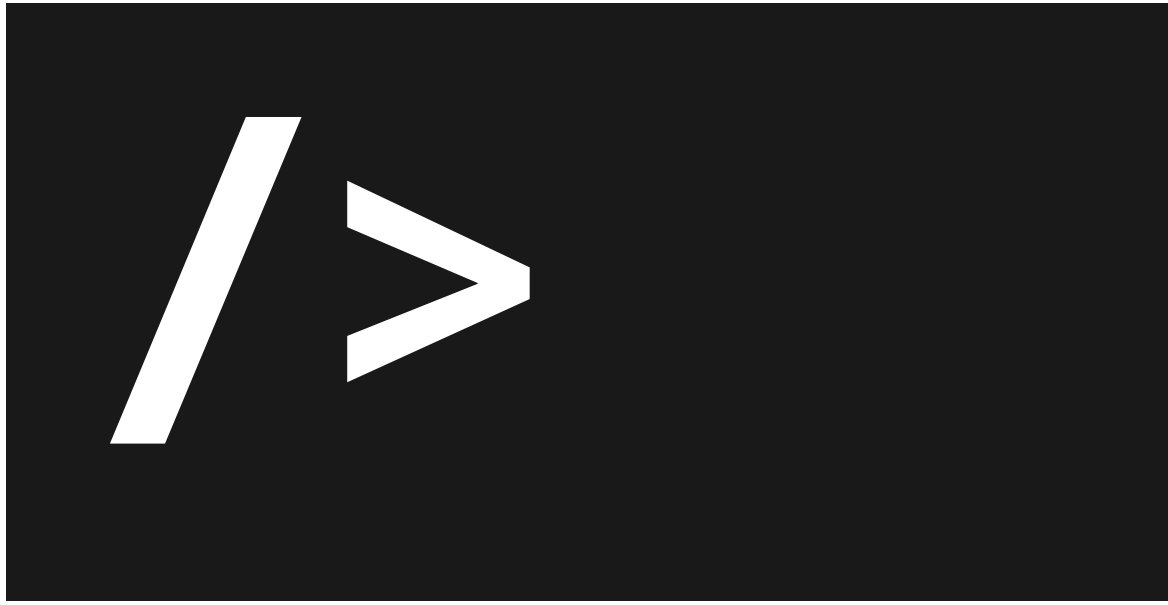
Uložení řetězcové hodnoty

- Text, serializovaný objekt (JSON), HTML, velikost do 512 MB… SET [key] [value]
- Užitečný je například GETSET [key] [value] (vrací hodnotu před přepsáním)
- Lze získat i více hodnot pro více klíčů pomocí MGET [key] [key] …

```
IDatabase redis = _connection.GetDatabase();
RedisValue date = redis.StringGet("datum");
bool result = redis.StringSet("datum", dt.ToString(), TimeSpan.FromSeconds(10);
```

DEMO

Output Cache Provider



Pipelining

- Redis podporuje pipelining při odeslání více asynchronních požadavků
- Dochází k multiplexaci, nečeká se na vyřešení jednoho příkazu aby odešel další
- Redukce latence a zvýšení výkonnosti

```
IDatabase redis = _connection.GetDatabase();

var task1 = redis.StringGetAsync("customer:1");
var task2 = redis.StringGetAsync("customer:2");

var customer1 = redis.Wait(task1);
var customer2 = redis.Wait(task2);
```

```
c:\Program Files\Redis>redis-benchmark -t set
SET: 123915.74 requests per second
GET: 120918.98 requests per second


c:\Program Files\Redis>redis-benchmark -t set
SET: 598802.44 requests per second
GET: 729927.06 requests per second
```

Použití: Čítače, počítadla návštěv

- Uložený řetězec může být číslo, nad kterým lze provádět dodatečné operace
- Nepotřebujeme aktuální hodnotu, pouze řekneme co má redis udělat s tou uloženou

```
IDatabase redis = _connection.GetDatabase();
long counter1 = redis.StringIncrement("user:1:visits");
```

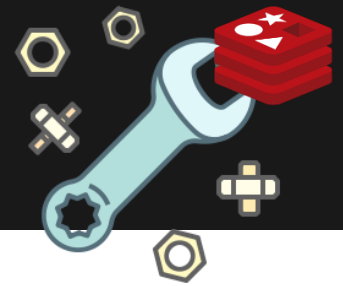
- Lze také inkrementovat o určitou hodnotu

```
long counter = redis.StringIncrement(key, 5);
```

- Pokud nás nezajímá výsledek, nemusíme na něj čekat

```
redis.StringIncrement(key, 1, CommandFlags.FireAndForget);
```

Použití: Indexování, vyhledávání, autocompletion



- Díky vysoké rychlosti lze použít redis cache k vytvoření indexů
- Například pokud vyhledáváme v pomalém úložišti na klíčové slovo „somekey“,

  můžeme si ID výsledků uchovat pro pozdější hledání.

```
List<Person> data = Database.Articles.Where(x => x.Text.Contains(value)).ToList();
redis.StringSet("search:" + value,
    JsonConvert.SerializeObject(data.Select(x => x.Id).ToArray()));

...

RedisValue searchResult = redis.StringGet("search:" + value);
List<int> ids = JsonConvert.DeserializeObject<List<int>>(searchResult);
List<Person> data2 = Database.All.Where(x => ids.Contains(x.Id)).ToList();
```

Použití: Uchování statistik objektů

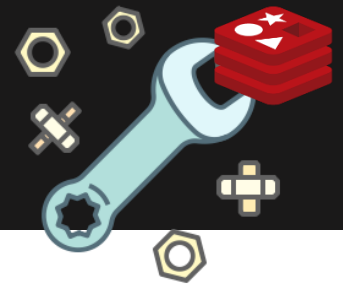
- Redis připouští jako hodnotu kolekci vlastností - HASH
- Místo ukládání single value klíčů user:1:visits lze uložit user:1 a visits mít jako vlastnost
- Nad vlastnostmi lze provádět různé operace

```
redis.HashSet("user:1", new[]
{
    new HashEntry("visits", 0), new HashEntry("likes", 1)
});

...

redis.HashIncrement("user:1", "visits"); // přidá návštěvu
HashEntry[] stats = redis.HashGetAll("user:1"); // vrátí visits a likes
```

Použití: Fronty a zásobník



- Hodnota může být kolekce stringů seřazená dle vložení - LIST
- Interní implementace jako LinkedList poskytuje přístup na head / tail s *O*(1)

```
long count = redis.ListRightPush("mail:welcome", "jmeno1@prijmeni.cz");
     count = redis.ListRightPush("mail:welcome", "jmeno2@prijmeni.cz");
     count = redis.ListRightPush("mail:welcome", "jmeno3@prijmeni.cz");

...

string mail = redis.ListLeftPop("mail:welcome");
```

Použití: Operace s množinami unikátních prvků

- Hodnota může být kolekce unikátních stringů - SET
- Velmi rychlé provádění Add/Remove a kontroly existence prvku
- Poskytuje množinové operace, např.: průnik / rozdíl

```
redis.SetAdd("p:1:tags", "novinky");
redis.SetAdd("p:2:tags", "elektro");

...

RedisValue[] tags = redis.SetMembers("p:1:tags");
RedisValue randomTag = redis.SetRandomMember("p:1:tags");
bool existsTag = redis.SetContains("p:1:tags", "elektro");
RedisValue[] commonTags =
    redis.SetCombine(SetOperation.Union, new RedisKey[] { "p:1:tags", "p:2:tags" });
```

Použití: Leaderboard

- Hodnota může být SET, kde každý prvek má své SCORE, tzv. SORTEDSET
- Kolekce je seřazena dle SCORE a nadtouto vlastností lze různě operovat

```
redis.SortedSetAdd("top", "884", 76);
redis.SortedSetAdd("top", "291", 45);
redis.SortedSetAdd("top", "198", 43);

double score = redis.SortedSetIncrement("top", "884", 1);
RedisValue[] top10 = redis.SortedSetRangeByRank("top", 1, 10);
RedisValue[] topByScore = redis.SortedSetRangeByScore("top", 800, 1000);
```

- I zde máme k dispozici operace průniku nebo rozdílu
  - Můžeme například udělat průnik objektů a sečíst score dle skupin

Použití: Message broker

- Implementace Publish / Subscribe patternu
- Publisher publikuje zprávy, které dostávají Subscribers
- Není zaručeno, že zpráva skutečně dorazí kam má

```
ISubscriber subscriber = _connection.GetSubscriber();
subscriber.Subscribe("chat", (channel, json) => {
    var message = JsonConvert.DeserializeObject<Message>(json);
    Debug.WriteLine(message.Title);
});

...

subscriber.Publish("chat",
    JsonConvert.SerializeObject(new Message {Title = "Test"})
);
```

Miroslav Holec
Software & Cloud Architect
Microsoft MVP: Microsoft Azure
MCSD, MCSA

odkaz.me/redis

Takeaways

- Vysoce výkonné úložiště strukturovaných dat
- Open-Source, řada nástrojů: CLI, Redsmine
- Output Cache a Session State providers
- Mnoho praktických využití mimo cachování
- Řada operací nad specializovanými typy
- Azure Redis Redis as a service

Q&A