Digitální informace

Digitalizace. Digitální zařízení. Základní technické vybavení počítače. Uspořádání paměti RAM v počítači. Zásobník a halda. Hodnotový a referenční datový typ. GarbageCollector. Jednotky informace. Číselné soustavy.

# Digitalizace:

Proces převodu informace do čísel (ve dvojkové soustavě), kterou umožňuje zpracovat počítač či další digitální zařízení.

Digitalizovaný signál se dá dobře přenášet, uchovávat a neomezeně reprodukovat bez ztráty informace. Ve stejném digitálním formátu lze také zobrazovat informace znakové (text), zvuk i obraz a různě je kombinovat.

# Informace:

* Textová

Kódování textů:

ASCII 1 znak ≅ 8 b

UNICODE 1 znak ≅ 16 bitů (65 536 znaků)  
současná celosvětová znaková sada, používaná v současných OS

Příklad:

*Jedna strana papíru formátu A4 obsahuje běžně cca 30 řádků po 100 znacích. Dokud byl jeden znak jeden Bajt, zabrala 3 000 B (Bajtů), tedy 3 KB (kilobajty).*

*Dnes se používají k zakódování jednoho znaku 2B/znak, stránka textu má cca 3 000 znaků, zabere tedy 2B/znak × 3 000 znaků = 6 000 B, tedy 6 KB.*

*Tisícistránková kniha by tedy zabrala cca 6 000 KB, tedy 6 MB*

Komprese textu – vždy bezztrátová

* grafická

digitalizace pomocí skeneru, fotoaparát

kódování barvy – barevné modely (RGB, CMYK, HSB)

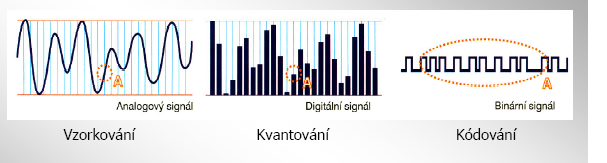
komprese ztrátová (jpg) nebo bezztrátová

* analogový signál (zvuk)

Pořízení [digitálního](http://cs.wikipedia.org/wiki/Digit%C3%A1ln%C3%AD) záznamu signálu lze rozdělit na tři fáze:

* *Vzorkování*

V definovaných časových intervalech je odebírán vzorek signálu, tím se ze spojitého časového průběhu stane diskrétní časový průběh

* *Kvantování*

Vzorky získané vzorkováním se pomocí [analogově-digitálního převodníku](http://cs.wikipedia.org/wiki/A/D_p%C5%99evodn%C3%ADk) převedou do číslicové formy.

* *Kódování*

Číslo získané kvantováním se kóduje z důvodu menší paměťové náročnosti, jednoduššího zpracování, přenosu

Digitalizací se nevratně ztrácí část informace obsažené v původním signálu, to se při zpětné rekonstrukci projeví jako [šum](http://cs.wikipedia.org/wiki/%C5%A0um).

*Příklad:*

*1 minuta zvuku se vzorkovací frekvence 44,1 kHz a 16bitový stereofonní záznam zabere jedna minuta záznamu 44100 · 16 · 2 · 60 : 8 = 10 584 000 bajtů.*

Velký objem dat => ztrátová komprese založena na „psychoakustickém“ modelu (formáty [MP3](http://cs.wikipedia.org/wiki/MP3), [Vorbis Ogg](http://cs.wikipedia.org/wiki/Vorbis), [WMA](http://cs.wikipedia.org/wiki/WMA) a [AAC](http://cs.wikipedia.org/wiki/AAC))

FLAC je kodek pro formát, který využívá bezztrátovou kompresi

* Video

Kombinace zvuku a sekvence obrázků

### Uspořádání paměti RAM v počítači:

<https://www.itnetwork.cz/csharp/oop/c-sharp-tutorial-referencni-a-hodnotove-tyoy-garbage-collector>

Zásobník i halda se nacházejí v paměti RAM. Rozdíl je v přístupu a velikosti. Halda je prakticky neomezená paměť, ke které je však přístup složitější a tím pádem pomalejší. Naopak zásobník je paměť rychlá, ale velikostně omezená.

* Zásobník (Stack)  
  všechny hodnotové typy (int, float, char, …) se vytvářejí na zásobníku  
  při volání metody jsou parametry uloženy na vrchol zásobníku, když volání metody končí, jsou ze zásobníku odstraněny
* Halda (Heap)

String, objekty,  
pro referenční datové typy se vyhrazuje souvislý blok z volné oblasti systémové paměti - haldy (angl. heap), která je spravována samostatným procesem GarbageCollector, samotný odkaz je také na zásobníku

## Referenční hodnoty v C# v paměti počítačeGarbage Collection

Mezi výhody programování v C# patří, že se programátoři nemusí podrobně starat o správu paměti, protože čištění paměti automaticky zajišťuje automatická správa paměti. Prochází paměť a hledá objekty, které už nejsou používány. Likviduje tyto objekty a uvolňuje zdroje držené těmito objekty.

Garbage collector je proces, který běží paralelně s naší aplikací, v samostatném vlákně. Zapnutí a vypnutí GC můžeme dokonce z kódu ovlivnit, i když to není v 99 % případů vůbec potřeba. Protože je jazyk řízený a nepracujeme s přímými pointery, není vůbec možné paměť nějak narušit, nechat ji přetéct a podobně, interpret se o paměť automaticky stará. Ztráta výkonu je minimální.

* snižuje chybovost (čili zvyšuje spolehlivost) kódu
* odstraňuje fragmentaci paměti (ihned po uvolnění všech nepoužívaných objektů zkomprimuje automatická správa paměti haldu tak, že přesune všechny zbývající objekty, aby tvořily jediný souvislý blok paměti)
* zjednodušuje zdrojový kód (vedlejší efekt)