

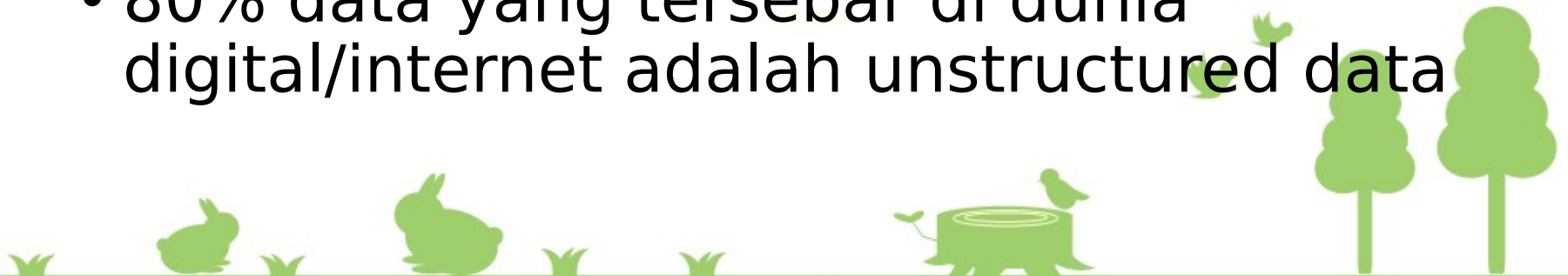
# Hadoop (1)

Hadoop Administration dan  
Hadoop Distribution File System  
(HDFS)



# Kenapa Hadoop?

- Dunia digital yang mengarah ke Big Data
- Mulanya, data masih bisa diproses oleh PC  
Mudah diproses, ukuran tidak terlalu besar  
Seiring waktu, data bertambah besar
- Data dan operasinya tidak bisa ditangani resource hardware yang ada
- 80% data yang tersebar di dunia digital/internet adalah unstructured data



# Hadoop (1)



- Open source project dari Apache Foundation
- Sebuah framework yang dibuat menggunakan Java untuk distributed computing dan penyimpanan data yang reliabel dan skalabel
- Dikembangkan oleh Doug Cutting pada tahun 2005 untuk Nutch
- Didanai oleh Yahoo dan pada tahun 2006 dihibahkan kepada Apache



# Hadoop (2)

- Yahoo masih pendana terbesar pada project Hadoop
- Menggunakan teknologi MapReduce milik Google
- Solusi pemrosesan dan analisis data yang berukuran sangat besar
- Dioptimalkan untuk menangani jumlah data masif
  - Structured
  - Unstructured
  - Semi Structured

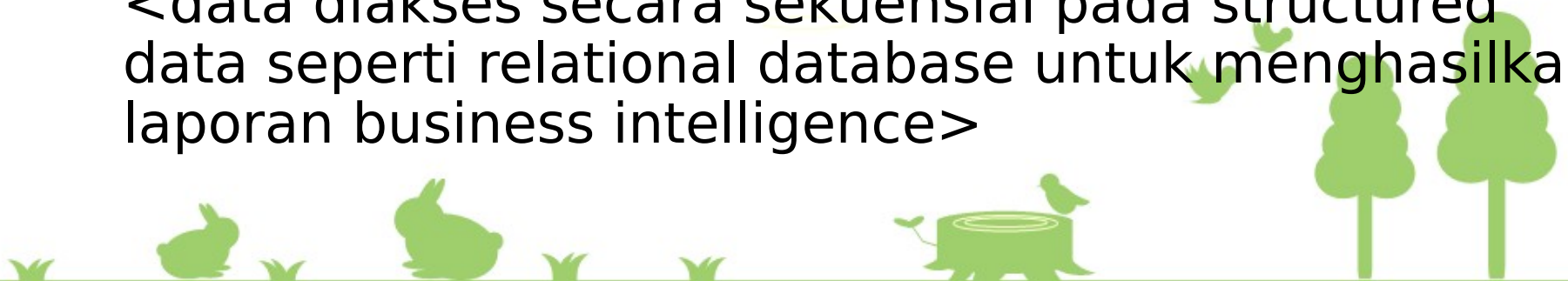


# Hadoop (3)

- Menggunakan resource hardware yang moderat
- Performa cepat dengan menggunakan paralel processing
  - \*Catatan: Hadoop menggunakan batch operation, untuk ukuran data yang masif → response time lambat → tidak bisa update instan, namun bisa menambahkan data
- Case: Jika data tidak konsisten, maka Hadoop mereplikasi data ke semua komputer yang ada dan jika 1 mati/hilang, data akan diproses di komputer lainnya

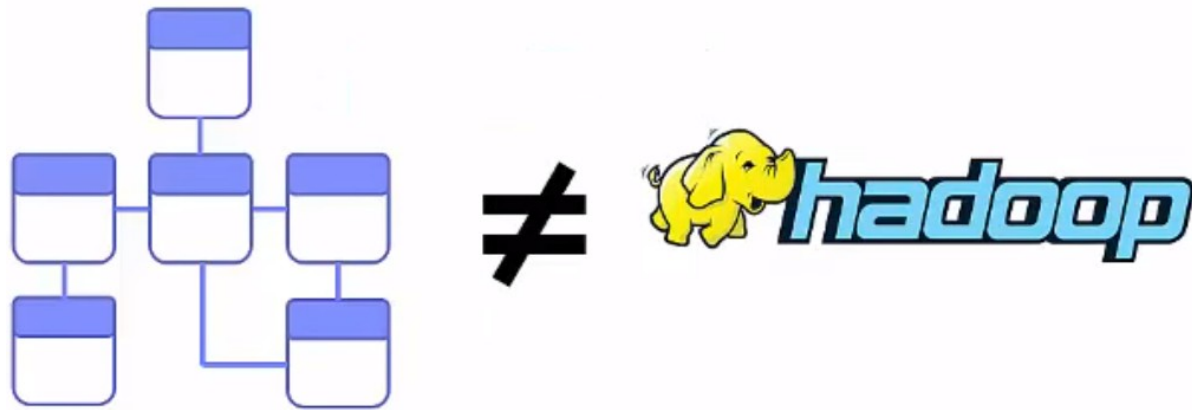
# Hadoop pada Proses

- Tidak cocok untuk OnLine Transaction Processing (OTLP)  
<data diakses secara random pada structured data seperti relational database>
- Tidak cocok untuk OnLine Analytical Processing (OLAP) atau Decision Support System (DSS)  
<data diakses secara sekuensial pada structured data seperti relational database untuk menghasilkan laporan business intelligence>



# Hadoop pada Proses (Lanjutan)

- Komplemen (Pelengkap) dari OTLP dan OLAP, bukan pengganti relational database



# Hadoop pada Data

- Tidak cocok untuk data yang saling berkterkaitan (tidak bisa diparalelisasikan karena tidak independen)
- Tidak cocok untuk akses data low latency
- Tidak cocok untuk banyak data berukuran kecil





# Big Data dengan Hadoop?

- Salah satu solusi Big Data
- Banyak solusi lain selain Hadoop
- Hadoop dapat diintegrasikan dengan layanan solusi analitik lainnya

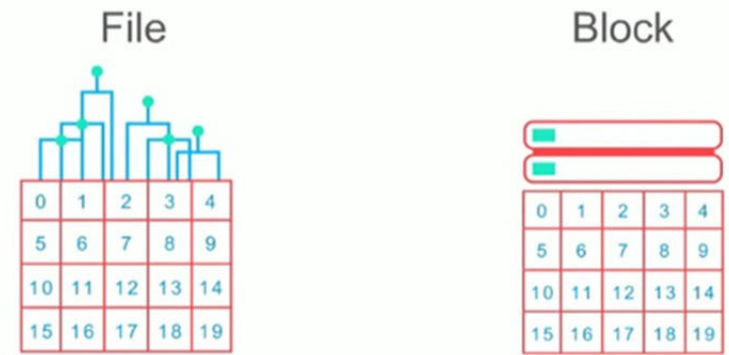


# Hadoop pada Cloud

- Pasangan yang sangat baik untuk Hadoop
- Fleksibel, simpel, dan skalabel untuk membuat Hadoop cluster (on-demand, dipakai saat dibutuhkan)
- Biaya lebih murah karena tidak berinvestasi pada hardware



# HDFS



- Pencarian lokasi data sangat mahal yang berguna ketika hanya untuk menganalisis sebagian kecil dataset
- Karena Hadoop bekerja di keseluruhan dataset menggunakan file berukuran besar
- Tidak random access sekuensial (mencari dari awal blok) lebih sedikit pencarian data
- Cocok untuk data streaming/sekuensial
- Menggunakan blocks untuk menyimpan sebuah file atau bagian dari sebuah file

# HDFS File Blocks

- Tidak sama dengan file block pada OS
- Default ukuran Hadoop Block 64MB ~128MB (rata-rata 128MB ~ lebih)
- Ukuran sebuah file bisa lebih besar dari 1 disk pada cluster – 1 file dibagi ke beberapa blok dan disebar ke beberapa node
- Jika sebagian file lebih kecil dari ukuran block hanya ruang yang dibutuhkan yang digunakan
- Block bekerja baik dengan replikasi

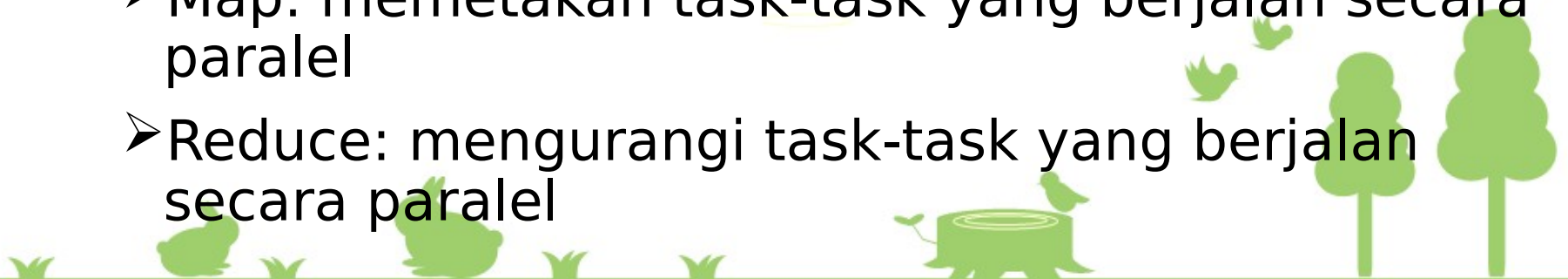
**Contoh sebuah file  
450MB**

128 MB	128 MB	128 MB	66M B
-----------	-----------	-----------	----------

# Framework MapReduce

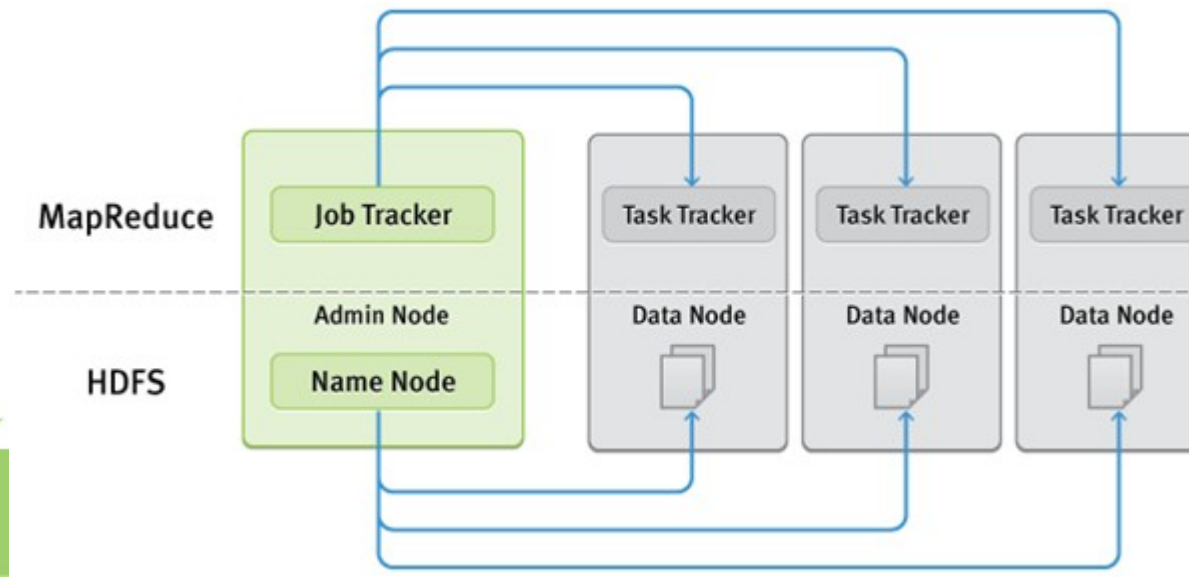
**\*dibahas lebih lanjut pada sesi berikutnya**

- Berbasis Google MapReduce
- Memproses dataset besar untuk beberapa jenis masalah yang dapat didistribusikan menggunakan banyak node
- Terdiri dari 2 transformasi: map dan reduce  
Dapat berjalan secara paralel distributed processing
  - Map: memetakan task-task yang berjalan secara paralel
  - Reduce: mengurangi task-task yang berjalan secara paralel



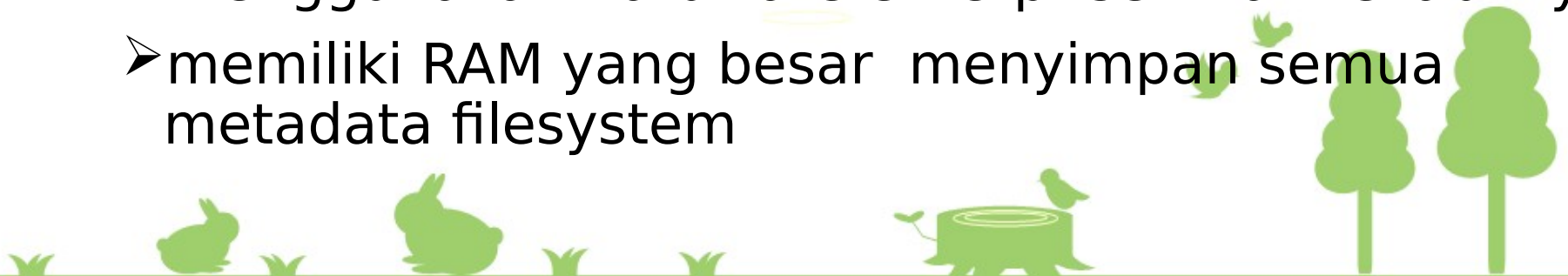
# Tipe-tipe Node pada Hadoop Cluster

- Dibagi menjadi 2, node HDFS atau MapReduce v1
  - HDFS NameNode dan DataNode
  - MapReduce v1 JobTracker dan TaskTracker
- Ada node HDFS lain:
  - Secondary NameNode
  - Checkpoint
  - Backup



# NameNode

- Hanya 1 per cluster
- Memiliki file beberapa DataNode berisi block-block data
- Mengelola namespace sistem file dan metadata
- NameNode single point of failure
  - sebaiknya direplikasi/mirror fisik
  - menggunakan hardware enterprise max reliability
  - memiliki RAM yang besar menyimpan semua metadata filesystem



# DataNode

- Banyak per cluster
- Blocks dari file berbeda bisa disimpan DataNode yang sama
- Mengelola blocks data dan melayani request client
- Update ke NameNode secara periodik, daftar blocks tersimpan
- Tidak perlu hardware enterprise dan replikasi hanya software





# JobTracker

- Mengelola MapReduce jobs
- Hanya 1 per cluster
- Menerima tugas dari client scheduling Map dan Reduce TaskTracker yang tepat (data disimpan <rack-aware manner>)
- Monitoring task gagal yang perlu direschedule pada TaskTracker berbeda



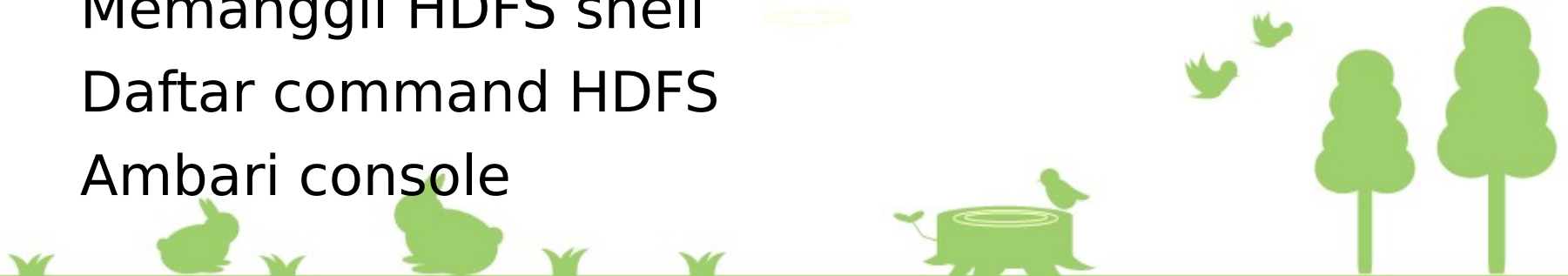
# TaskTracker

- Banyak TaskTracker per cluster → paralelisme task map dan reduce
- Tiap TaskTracker → call Java VM → run task map atau reduce
  - Berkomunikasi → heartbeat message → JobTracker
  - Membaca blocks dari DataNode



# HDFS Command Line

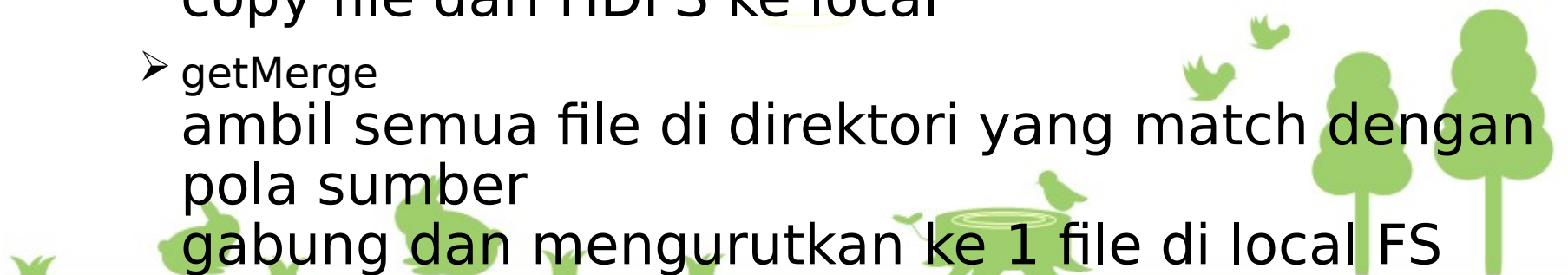
Memanggil HDFS shell  
Daftar command HDFS  
Ambari console



# HDFS file command interface

- Panggil FileSystem (FS) shell  
`hdfs dfs <args>`
- Contoh command daftar isi direktori saat ini di HDFS  
`hdfs dfs -ls`
- FS shell command path URI (Uniform Resource Identifier)  
argumen  
`scheme://authority/path`
- Scheme: `hdfs` `hdfs` , local filesystem file  
`hdfs dfs -cp (contoh command copy dari file ke HDFS)`  
`file:///sampleData/spark/myfile.txt hdfs://rvm.svl.ibm.com:8020/user/spark/test/myfile.txt`
- Scheme dan authority optional, default dari *core-site.xml* conf file
- Mayoritas command FS shell mirip command UNIX

# HDFS file command interface

- Meski bukan POSIX compliant, ada beberapa command seperti POSIX  
cat, chgrp, chmod, chown, cp, dua, ls, mkdir, mv, rm, stat, tail
  - Command spesifik HDFS  
copyFromLocal, copyToLocal, get, getmerge, put, setrep
    - copyFromLocal / put  
copy file dari local FS ke HDFS
    - copyToLocal / get  
copy file dari HDFS ke local
    - getMerge  
ambil semua file di direktori yang match dengan pola sumber  
gabung dan mengurutkan ke 1 file di local FS
- 

➤ setRep

- ✓ set faktor replikasi sebuah file
- ✓ dapat dieksekusi rekursif untuk mengganti satu tree
- ✓ dapat dispesifikkan untuk menunggu hingga level replikasi terpenuhi



# Hadoop Administration

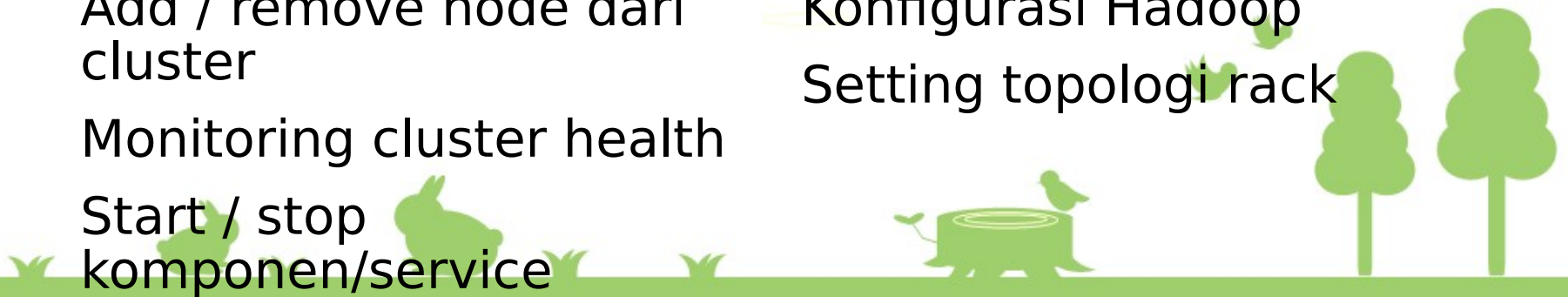
Add / remove node dari cluster

Monitoring cluster health

Start / stop komponen/service

Konfigurasi Hadoop

Setting topologi rack



# Add/Remove Node dari Cluster

- Dapat dilakukan dari Ambari Console
  - butuh IP address atau hostname dari node
  - node harus reachable (komunikasi antara master dan child)
- \* /etc/hosts di master dan child harus di-update
- ✓ Ambari console tab Hosts
- ✓ Actions Add New Hosts
- ✓ Dialog tambah 1 node atau lebih masukkan IP address atau hostname atau keduanya (bisa range IP atau regular expression hostname)
- ✓ Add Multiple Service (checkbox)
  - \*service dapat di-remove
- Remove node, matikan service terlebih dahulu



# Monitoring Cluster Health

- Dapat dilakukan melalui Ambari Console Monitor node dan service yang ada
- Disk Space
  - DFS Disk Check dengan DFS Report (`hdfs dfsadmin -report`)
    - mengetahui low storage
    - dapat dilihat di Ambari Console



# Start/Stop Komponen

- Tidak semua komponen harus berjalan  
Stop beberapa komponen hemat resource
- Service dapat di-start/stop dari Main Dashboard Ambari Console
- Sebaiknya tidak menyalakan semua service dalam satu waktu



# Configuration files

- `hadoop-env.sh` – environment variable untuk run Hadoop
- `core-site.xml` – configuration setting Hadoop Core, I/O setting HDFS dan MapReduce
- `hdfs-site.xml` – configuration setting HDFS daemon: NameNode, Secondary NameNode, DataNode
- `mapred-site.xml` – configuration setting MapReduce daemon: JobTracker, TaskTracker
- `Masters` – Daftar mesin yang run Secondary NameNode
- `slaves` – Daftar mesin yang run DataNode dan TaskTracker
- `hadoop-metrics.properties` – kontrol metric pada Hadoop
- `log4j.properties` – system logfiles, NameNode audit log, task log untuk TaskTracker child proses

# hadoop-env.sh settings

- Sebagian besar variabel → default tidak diset
- Hanya `export JAVA_HOME` harus diset ke Java SDK
- `HADOOP_HOME` – berisi node dan config file  
`/usr/iop/current/hadoop-client`
- `HADOOP_LOG_DIR` – menjaga log `/var/log/Hadoop/$USER`
- `HADOOP_HEAPSIZE` – digunakan JVM untuk tiap daemon
  - NameNode - `HADOOP_NAMENODE_OPTS`
  - DataNode - `HADOOP_DATANODE_OPTS`
  - Secondary NameNode - `HADOOP_SECONDARYNAMENODE_OPTS`
  - JobTracker - `HADOOP_JOBTRACKER_OPTS`
  - TaskTracker - `HADOOP_TASKTRACKER_OPTS`
- Environment variable lain - `HADOOP_CLASSPATH`, `HADOOP_PID_DIR`

# core-site.xml setting

- `fs.defaultfs` – nama default filesystem. URI dengan scheme dan authority menentukan implementasi FileSystem. Scheme menentukan config property (`fs.SCHEME.impl`) penamaan class implementasi FileSystem. Authority menentukan host, port, dll FileSystem. Default file:///
- `hadoop.tmp.dir` – base untuk direktori sementara lainnya `/tmp/hadoop-${user.name}`
- `fs.trash.interval` – jumlah menit antara trash checkpoint. Jika 0, trash feature disabled (default). Jika  $> 0$ , file terhapus akan dimasukkan ke `.trash` pada direktori home milik user
- `io.file.buffer.size` – ukuran buffer untuk sequence file. Kelipatan hardware page size (4096 pada x86), menentukan buffer saat read dan write

# core-site.xml setting

- `hadoop.rpc.socket.factory.class.default` – default SocketFactory untuk digunakan. Parameter untuk diformat sebagai `package.FactoryClassName`
- `hadoop.rpc.socket.factory.class.clientprotocol` – SocketFactory untuk koneksi ke DFS. Jika null atau kosong, gunakan `hadoop.rpc.socket.class.default`. Digunakan juga oleh DFSClient untuk membuat socket ke DataNode

\*biarkan kedua parameter tersebut kosong, tandai FINAL

# hdfs-site.xml setting

- `dfs.datanode.data.dir` – menentukan di mana DFS datanode harus menyimpan blocks-nya pada localFS
- `dfs.namenode.name.dir` - menentukan di mana DFS namenode harus menyimpan name table-nya pada localFS
- `dfs.blocksize` – HDFS blocksize, default 64MB.  
Rekomendasi: set ke 128MB atau sesuai dengan ukuran data



# mapred-site.xml configuration (1)

- `mapreduce.jobtracker.hosts` – menamai file yang berisi daftar nodes yang terhubung dengan jobtracker. Jika value-nya kosong, semua host diizinkan
- `mapreduce.jobtracker.hosts.exclude` – menamai file yang berisi daftar hosts yang harus di-exclude oleh jobtracker. Jika value-nya kosong, tidak ada host yang di-exclude
- `mapreduce.job.maxtaskfailures.per.tracker` – jumlah task-failure pada tasktracker job yang diberikan setelah task baru mana yang tidak di-assign ke job. Default 3.
- `mapreduce.jobtracker.tasktracker.maxblacklists` – jumlah blacklist untuk TaskTracker oleh berbagai job yang di-blacklisted di semua job. Tracker akan diberikan task kemudian (setelah 1 hari). Tracker akan menjadi healthy tracker setelah restart. Default 4.



# mapred-site.xml configuration (2)

- `mapreduce.job.reduces` – jumlah default reduce task per job. Umumnya diset ke 99% ke kapasitas reduce cluster, jadi jika sebuah node gagal/down, makan reduce masih bisa dieksekusi dalam 1 gelombang. Dihiraukan ketika `mapred.job.tracker` “local”. Default 1. Rekomendasi diset 90%.
- `mapreduce.map.speculative` – Jika TRUE, multiple instance dari beberapa map task akan dieksekusi paralel. Default TRUE.
- `mapreduce.reduce.speculative` – Jika TRUE, multiple instance dari beberapa reduce task akan dieksekusi paralel. Default TRUE. Rekomendasi FALSE.
- `mapreduce.tasktracker.map.tasks.maximum` - Jumlah max map tasks yang akan dijalankan secara simultan oleh TaskTracker. Default 2. Rekomendasi set relevan ke jumlah CPU dan memori di tiap DataNode

# mapred-site.xml configuration (3)

- `mapreduce.tasktracker.reduce.tasks.maximum` - Jumlah max reduce tasks yang akan dijalankan secara simultan oleh TaskTracker. Default 2. Rekomendasi set relevan ke jumlah CPU dan memori di tiap DataNode
- `mapreduce.jobtracker.taskscheduler` – Class bertanggung jawab untuk scheduling task. Default ke FIFO scheduler. Rekomendasi menggunakan Job Queue Task – `org.apache.hadoop.mapred.JobQueueTaskScheduler`
- `mapreduce.jobtracker.restart.recover` – Recover job gagal ketika JobTracker restart. Untuk cluster produksi direkomendasikan diset TRUE.



# mapred-site.xml configuration (4)

- `mapreduce.cluster.local.dir` - Direktori lokal di mana MapReduce menyimpan file data intermediate. Bisa berbentuk daftar terpisah dengan koma, direktori di divais berbeda untuk menyebar I/O disk. Direktori yang tidak eksis dihiraukan. Default `${hadoop.tmp.dir}/mapred/local`

