

```

class gov.nasa.jpjf.Config {
    Object getInstance(key,type) throws Config.Exception
    Object getEssentialInstance(key,type)..
    boolean getBoolean(key) ..
}

```

## Config object

```

..
myheuristic.some_value=42
search.class=..HeuristicSearch
search.heuristic.class=MyHeuristic
vm.class=..JVM
..

```

fixed type

fixed+configured types

JPF

Search

Heuristic

...

VM

Scheduler

...

## initialization

class: `init(config)`  
inst: `ctor(config)`

Config.Exception

- missing entry
- wrong type
- general exception

`java {-vm-arg..} gov.nasa.jpjf.JPF [-c config-file] {+key=value..} [-show] main-class {app-arg..}`

command line properties

+

mode properties

+

default properties

```

> java gov.nasa.jpjf.JPF -c bfs.properties
+search.heuristic.class=MyHeuristic
+myheuristic.some_value=42
MyTestApp

```

```

# breadth first JPF configuration
search.class = \
    gov.nasa.jpjf.search.heuristic.HeuristicSearch
search.heuristic.class = \
    gov.nasa.jpjf.search.heuristic.BFSHeuristic

```

```

# section 1: general properties
log = warning
..
# section 2: Search properties
search.class = gov.nasa.jpjf.search.DFSearch
..
# section 3: JVM properties
vm.class = gov.nasa.jpjf.jvm.JVM
..

```

## lookup

1. command-line specified file (`-c file`)
2. **jpjf.properties** in current dir
3. **jpjf.properties** in JPF root dir (can be set via `+jpjf.basedir=dir` command line option)
4. **jpjf.properties** resource in jar (loaded via `gov.nasa.jpjf.JPF`)

1. **default.properties** in current dir
2. **default.properties** in JPF root dir (can be set via `+jpjf.basedir=dir` command line option)
3. **default.properties** resource in jar (loaded via `gov.nasa.jpjf.JPF`)

## debug

`-show` prints used property values  
`+log.level=config` prints used file sources