Below are examples of highlighted blocks of code...

YAML

```yaml
filetype: diff

detect:
    filename: "\\.diff$"
    header: "^((---.*)|(\\+\\+\\+.*))"
rules:
    - statement: "(^\\+\\+\\+.*)"
    - statement: "(^---.*)"
    - type: "(^@@.*)"
    - constant.number: "(^\\+.*)"
    - preproc: "(^-.*)"
```

INI

```ini
[ctype]
; priority=20
extension=ctype.so
```

ping

```
PING jessex (127.0.1.1) 56(84) bytes of data.
64 bytes from jessex (127.0.1.1): icmp_seq=1 ttl=64 time=0.031 ms
64 bytes from jessex (127.0.1.1): icmp_seq=2 ttl=64 time=0.033 ms

--- jessex ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3033ms
rtt min/avg/max/mdev = 0.031/0.036/0.041/0.004 ms
```

df

```
Filesystem      Size  Used Avail Use% Mounted on
udev            7.7G     0  7.7G   0% /dev
tmpfs           1.6G  2.9M  1.6G   1% /run
/dev/nvme0n1p2   23G   20G  1.8G  92% /
tmpfs           7.7G   56M  7.7G   1% /dev/shm
tmpfs           5.0M  4.0K  5.0M   1% /run/lock
/dev/nvme0n1p6  434G  227G  185G  56% /home
```

Shell

```
#!/bin/sh -e
PATH="/sbin:/bin"
RUN_DIR="/run/network"
IFSTATE="$RUN_DIR/ifstate"
STATEDIR="$RUN_DIR/state"

[ -x /sbin/ifup ] || exit 0
[ -x /sbin/ifdown ] || exit 0

. /lib/lsb/init-functions

CONFIGURE_INTERFACES=yes
EXCLUDE_INTERFACES=
VERBOSE=no

[ -f /etc/default/networking ] && . /etc/default/networking

verbose=""
[ "$VERBOSE" = yes ] && verbose=-v

check_ifstate() {
    if [ ! -d "$RUN_DIR" ] ; then
 if ! mkdir -p "$RUN_DIR" ; then
     log_failure_msg "can't create $RUN_DIR"
     exit 1
 fi
 if ! chown root:netdev "$RUN_DIR" ; then
     log_warning_msg "can't chown $RUN_DIR"
 fi
    fi
    if [ ! -r "$IFSTATE" ] ; then
 if ! :> "$IFSTATE" ; then
     log_failure_msg "can't initialise $IFSTATE"
     exit 1
 fi
    fi
}
```

PHP

```php
<?php
if( !function_exists('mb_str_split')){
    function mb_str_split(  $string = '', $length = 1 , $encoding = null ){
        if(!empty($string)){
            $split = array();
            $mb_strlen = mb_strlen($string,$encoding);
            for($pi = 0; $pi < $mb_strlen; $pi += $length){
                $substr = mb_substr($string, $pi,$length,$encoding);
                if( !empty($substr)){
                    $split[] = $substr;
                }
            }
        }
        return $split;
    }
}
```

Golang

```golang
package main

import (
    "fmt"
    "os"
    "path/filepath"
)

func main() {
    ex, err := os.Executable()
    if err != nil {
        panic(err)
    }
    exPath := filepath.Dir(ex)
    fmt.Println(exPath)
}
```

C

```c
#include <stdio.h>
int main(int argc, char *argv[]) {
    printf("%s\n", "Hello world!");
}
```

XML

```
<?xml version="1.0" encoding="UTF-8"?>
<deck>
 <title>Sample Deck</title>
 <canvas width="1024" height="768"/>
 <slide bg="maroon" fg="white" duration="1s">
>   <image xp="20" yp="30" width="256" height="256" name="/home/jesse/docs/packman.io_logo.png"/
  <text xp="20" yp="80" sp="3" link="https://packman.io/">Deck uses these elements</text>
  <line xp1="20" yp1="75" xp2="90" yp2="75" sp="0.3" color="rgb(127,127,127)"/>
  <list xp="20" yp="70" sp="1.5">
   <li>canvas</li>
   <li>slide</li>
   <li>text</li>
   <li>list</li>
   <li>image</li>
   <li>line</li>
   <li>rect</li>
   <li>ellipse</li>
   <li>curve</li>
   <li>arc</li>
  </list>
  <line    xp1="20" yp1="10" xp2="30" yp2="10"/>
  <rect    xp="35"  yp="10" wp="4" hp="3" color="rgb(127,0,0)"/>
  <ellipse xp="45"  yp="10" wp="4" hp="3" color="rgb(0,127,0)"/>
  <curve   xp1="60" yp1="10" xp2="75" yp2="20" xp3="70" yp3="10" />
  <arc     xp="55"  yp="10" wp="4" hp="3" a1="0" a2="180" color="rgb(0,0,127)"/>
  <polygon xc="75 75 80" yc="8 12 10" color="rgb(0,0,127)"/>
 </slide>
</deck>
```