QUELQUES PISTES POUR SOIGNER SES TRACES D'EXÉCUTION 

Merci à nos sponsors

« Etoile »

« Flocon »

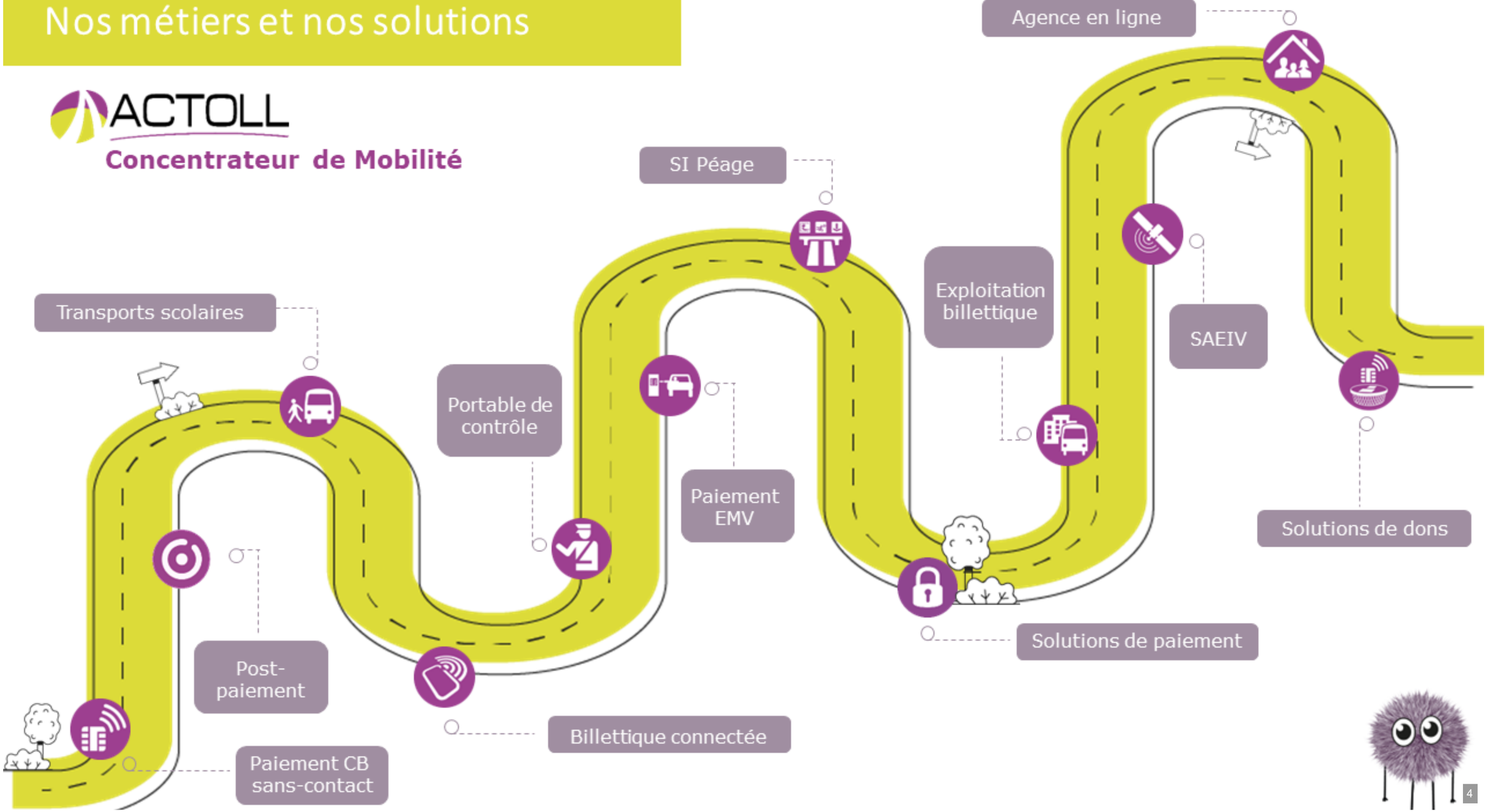
QUI SUIS-JE ?

- Initialement développeuse C/C++/C#
- Intégratrice, testeuse
- Support client
- DevRel in progress
- Twitter : @La_Fee_Dragee
- LinkedIn : https://www.linkedin.com/in/virginiecasavecchia/

Nos métiers et nos solutions

ACTOLL

Concentrateur de Mobilité



QUELQUES PISTES POUR SOIGNER SES TRACES D'EXÉCUTION 🤖⛷️


Benjamin Dauvissat - @bdauvissat@mastodon... @bdauvi... · 6 oct.
En réponse à @ponceto91
J'ai trouvé pire que l'absence de log.
Les logs hyper verbeux dans lesquels on ne trouve aucune information pertinente.
6
1
14

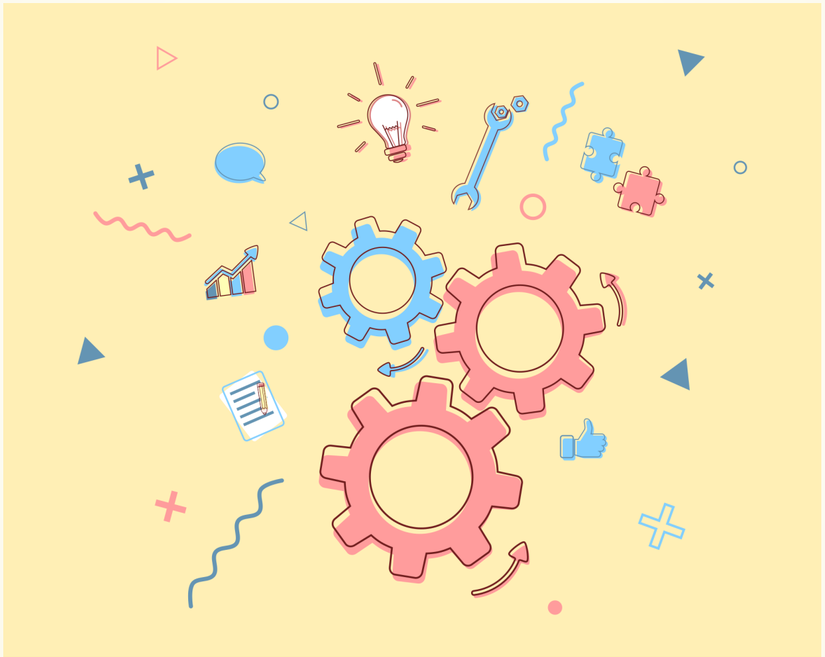
QUELQUES DÉFINITIONS

- SaaS (Software as a Service)
- Exploitant
  - Poste de contrôle péage

DISCLAIMER

A QUI/QUOI SERVENT LES TRACES ?

RÉALISER LE DÉVELOPPEMENT



VOUS AUSSI VOUS L'AVEZ DÉJÀ FAIT...

```
public List<Glycemie> importFichier() {
  logger.debug("importFichier début -----------------");
  /*
  ...
  */
  logger.debug("importFichier fin -----------------");
  return retour;
}
```

FACILITER LES TESTS ET LA CORRECTION D'ANOMALIES

FACILITER LES TESTS ET LA CORRECTION D'ANOMALIES

```
//...
catch (IOException e) {
    logger.error(+ "Une erreur est survenue lors du traitement du fichier "
                + fichier.toString() + ".");
    e.printStackTrace();
}
//...
```

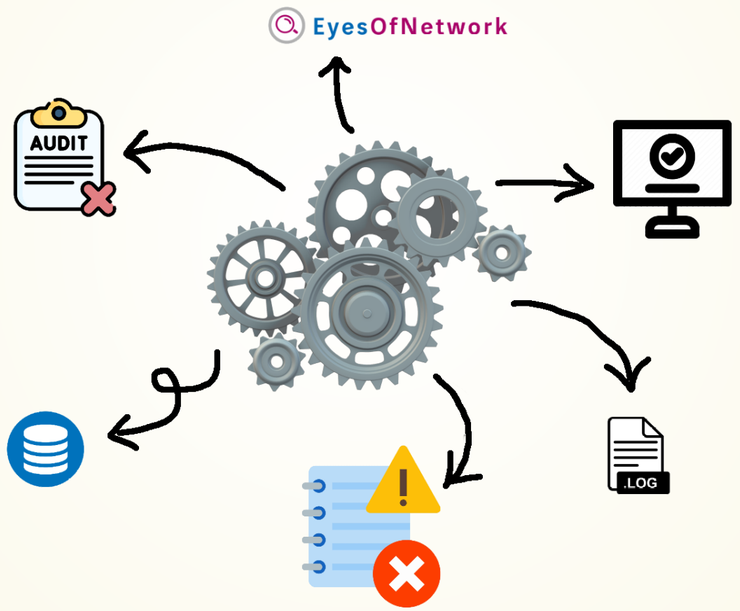
AIDER L'EXPLOITATION

AIDER L'EXPLOITATION

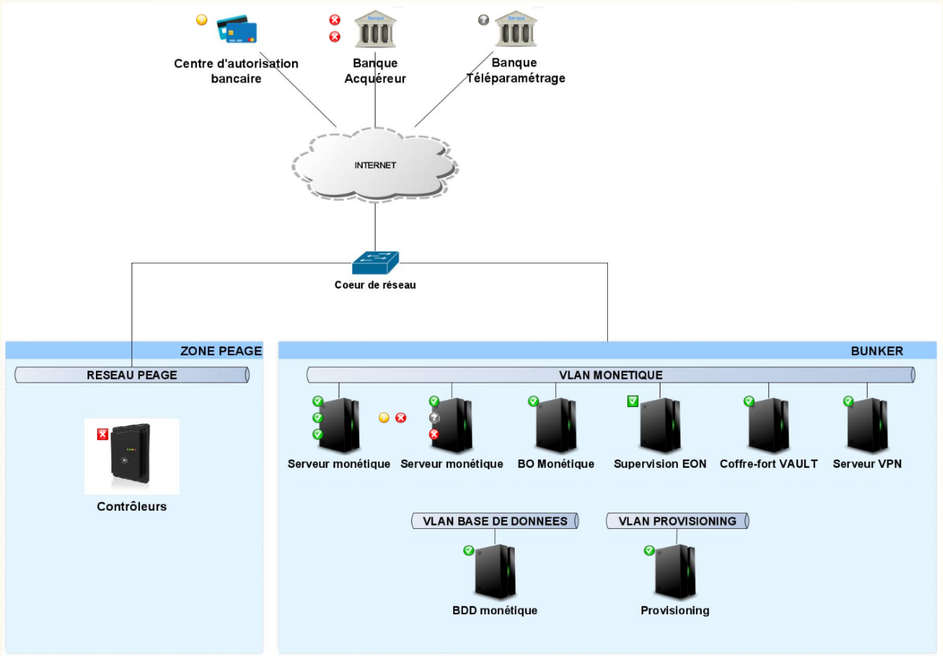
```
if (!repertoire.isDirectory()) {
    logger.error("\"" + repertoire.toString() + "\""
                + " n'est pas un répertoire. Arrêt du traitement.");

    return retour;
}
```

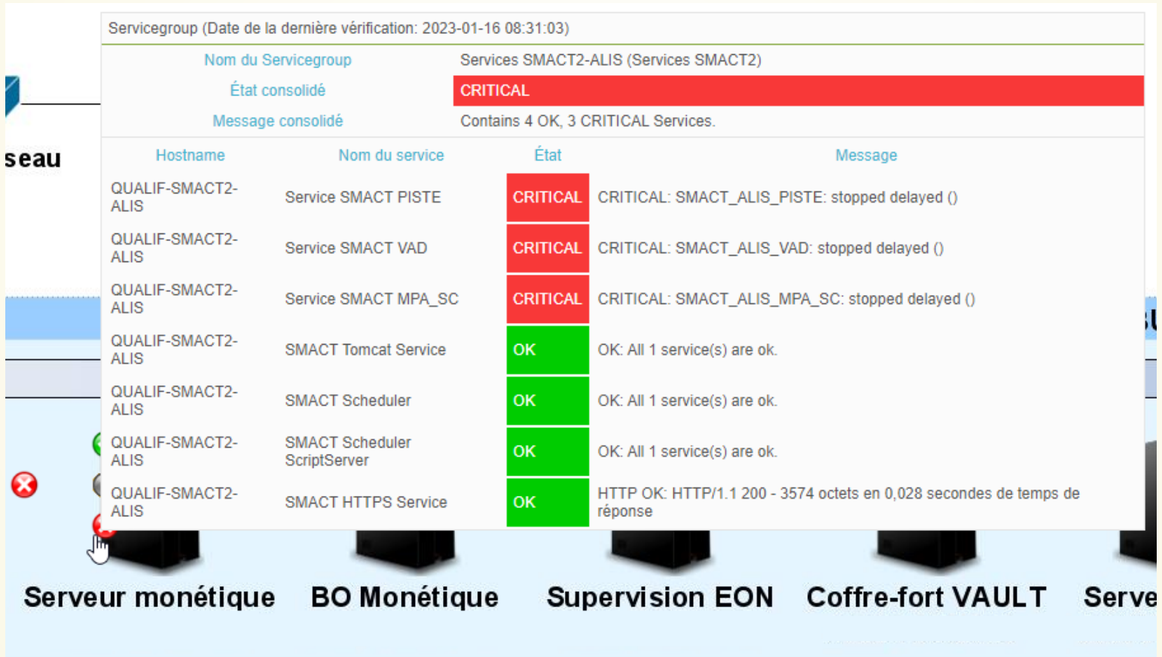
DIVERSES CIBLES



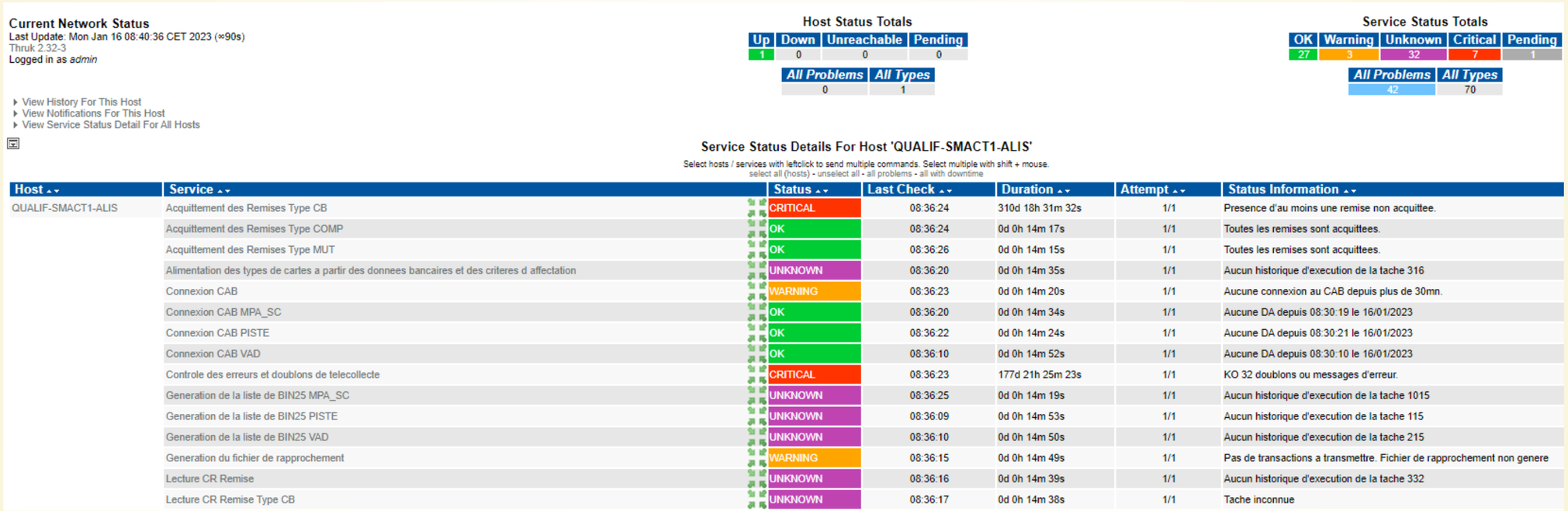
UN EXEMPLE DE SUPERVISION EXPLOITANT - NAGVIS



UN EXEMPLE DE SUPERVISION EXPLOITANT - NAGVIS



UN EXEMPLE DE SUPERVISION EXPLOITANT - NAGIOS EN VUE LISTE



Current Network Status
Last Update: Mon Jan 16 08:40:36 CET 2023 (∞90s)
Thruk 2.32-3
Logged in as *admin*

- View History For This Host
- View Notifications For This Host
- View Service Status Detail For All Hosts

Host Status Totals

Up	Down	Unreachable	Pending
1	0	0	0

All Problems	All Types
0	1

Service Status Totals

OK	Warning	Unknown	Critical	Pending
27	3	32	7	1

All Problems	All Types
42	70

Service Status Details For Host 'QUALIF-SMACT1-ALIS'

Select hosts / services with leftclick to send multiple commands. Select multiple with shift + mouse.
select all (hosts) - unselect all - all problems - all with downtime

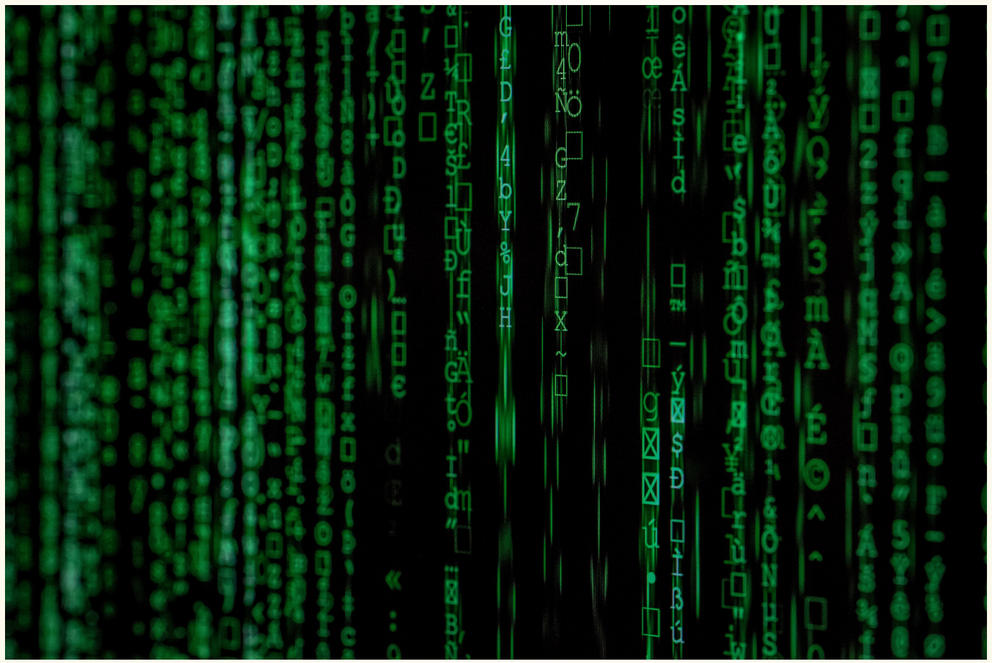
Host	Service	Status	Last Check	Duration	Attempt	Status Information
QUALIF-SMACT1-ALIS	Acquittement des Remises Type CB	CRITICAL	08:36:24	310d 18h 31m 32s	1/1	Presence d'au moins une remise non acquittee.
	Acquittement des Remises Type COMP	OK	08:36:24	0d 0h 14m 17s	1/1	Toutes les remises sont acquittees.
	Acquittement des Remises Type MUT	OK	08:36:26	0d 0h 14m 15s	1/1	Toutes les remises sont acquittees.
	Alimentation des types de cartes a partir des donnees bancaires et des criteres d affectation	UNKNOWN	08:36:20	0d 0h 14m 35s	1/1	Aucun historique d'execution de la tache 316
	Connexion CAB	WARNING	08:36:23	0d 0h 14m 20s	1/1	Aucune connexion au CAB depuis plus de 30mn.
	Connexion CAB MPA_SC	OK	08:36:20	0d 0h 14m 34s	1/1	Aucune DA depuis 08:30:19 le 16/01/2023
	Connexion CAB PISTE	OK	08:36:22	0d 0h 14m 24s	1/1	Aucune DA depuis 08:30:21 le 16/01/2023
	Connexion CAB VAD	OK	08:36:10	0d 0h 14m 52s	1/1	Aucune DA depuis 08:30:10 le 16/01/2023
	Controle des erreurs et doublons de telecollecte	CRITICAL	08:36:23	177d 21h 25m 23s	1/1	KO 32 doublons ou messages d'erreur.
	Generation de la liste de BIN25 MPA_SC	UNKNOWN	08:36:25	0d 0h 14m 19s	1/1	Aucun historique d'execution de la tache 1015
	Generation de la liste de BIN25 PISTE	UNKNOWN	08:36:09	0d 0h 14m 53s	1/1	Aucun historique d'execution de la tache 115
	Generation de la liste de BIN25 VAD	UNKNOWN	08:36:10	0d 0h 14m 50s	1/1	Aucun historique d'execution de la tache 215
	Generation du fichier de rapprochement	WARNING	08:36:15	0d 0h 14m 49s	1/1	Pas de transactions a transmettre. Fichier de rapprochement non genere
	Lecture CR Remise	UNKNOWN	08:36:16	0d 0h 14m 39s	1/1	Aucun historique d'execution de la tache 332
	Lecture CR Remise Type CB	UNKNOWN	08:36:17	0d 0h 14m 38s	1/1	Tache inconnue

COMMENT CHOISIR LE NIVEAU D'UNE TRACE ?

NIVEAUX COURAMMENT UTILISÉS

- VERBOSE
- DEBUG
- INFO
- WARNING
- ERROR

VERBOSE / DEBUG



VERBOSE / DEBUG - EXEMPLE

```
2023-01-15 17:42:57.439 [DEBUG ] Creating shared instance of singleton bean 'propertySourcesPlaceholderConfigurer'
2023-01-15 17:42:57.442 [DEBUG ] Creating shared instance of singleton bean 'org.springframework.context.event.internalE
2023-01-15 17:42:57.444 [DEBUG ] Creating shared instance of singleton bean 'org.springframework.context.event.internalE
2023-01-15 17:42:57.447 [DEBUG ] Creating shared instance of singleton bean 'org.springframework.context.annotation.inte
```

INFO

- 📝 Tâches métier
- 👷 Intelligible pour l'exploitant

INFO - EXEMPLE

```
2023-01-15 17:22:00.019 [INFO  ] Traitement du fichier D:\Documents\GitHub\glycescan\test_files\datafile_03.txt.
2023-01-15 17:22:00.019 [INFO  ] Déplacement du fichier "D:\Documents\GitHub\glycescan\test_files\datafile_03.txt" vers
2023-01-15 17:22:00.020 [INFO  ] Déplacement réussi.
```

WARNING

- ⚠️ Alerte
- 🥲 Fonctionnement dégradé
- 👀 Surveillance

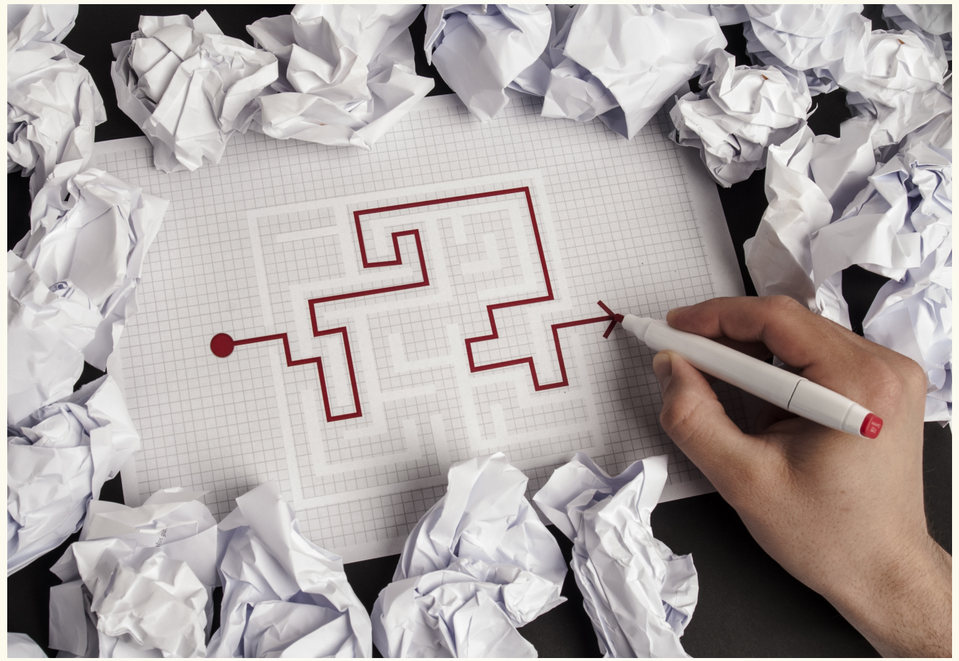
ERROR

- 🔥 Alerte grave
- 🚧 Fonctionnement en péril
- 🛠️ Action nécessaire

ERROR - EXEMPLE

```
2023-01-15 17:30:00.017 [ERROR ] "D:/bad_directory" n'est pas un répertoire. Arrêt du traitement. Veuillez vérifier la c
```

QUELLES INFORMATIONS INDIQUER ?



EN C/C++ UTILISEZ LES CONSTANTES PRÉPROCESSEUR ET COMPILATEUR !

```cpp
// File HelloThere.cpp
#include <iostream>

void general_kenobi()
{
  printf("WARN  : %s:%s\r\n\r\n",__FILE__,__func__);
  return;
}

int main(int argc, char *argv[])
{
  printf("\r\n");
  printf("INFO  : Quelques pistes pour soigner ses traces.\
  printf("DEBUG : %s(%d).\r\n",__FILE__,__LINE__);
  printf("DEBUG : %s\r\n",__PRETTY_FUNCTION__);
  // printf("DEBUG : %s\r\n",__FUNCSIG__);// Visual C++
  general_kenobi();
  return 0;
}
```

```
PS D:\Documents\GitHub\talks\HelloWorld> .\HelloThere.exe

INFO  : Quelques pistes pour soigner ses traces.
DEBUG : D:\Documents\GitHub\talks\HelloWorld\HelloThere.cpp(13
DEBUG : int main(int, char**)
WARN  : D:\Documents\GitHub\talks\HelloWorld\HelloThere.cpp:ge

PS D:\Documents\GitHub\talks\HelloWorld>
```

PLUS EN DÉTAIL

- Objet métier, identifiant (numéro de commande, nom de fichier...)
- Écarts valeur reçue / attendue
- Traitement métier en cours
- Actions possibles
  → dans les traces ou en annexe

PLUS EN DÉTAIL - EXEMPLE

```
2023-01-26 12:10:00.017 [ERROR ] Champ n°3 incorrect : "2023/01/26" n'est pas une date valide. Format attendu "dd/MM/yyy
2023-01-26 12:10:00.017 [ERROR ] Veuillez corriger le fichier "mon_fichier.txt" et réessayer.
```

ET LA RÉGLEMENTATION DANS TOUT ÇA ?

- PCI-DSS (Payment Card Industry Data Security Standard)
- CNIL (Commission Nationale de l'Informatique et des Libertés)
  - RGPD (Règlement général sur la protection des données)

Virginie CASAVECCHIA
@La_Fee_Dragee(Fairywen)

ACTOLL

UN EXEMPLE DANS DU VRAI CODE ?

Voyons un outil de lecture de fichier, codé en Java.

(Merci à Benjamin Dauvissat - Twitter @bdauvissat)

```java
package net.benji.glycescan.service.impl;

// import des dépendances

@Service("lectureFichierGlycemie")
public class LectureFichierGlycemieImpl implements ILectureFichierGlycemie {

    private static final Logger logger = LogManager.getLogger(LectureFichierGlycemieImpl.class);

    @Value("${local.path.glycemie}")
    private String path;

    @Value("${local.path.glycemie.done}")
    private String done;

    @Override
    public List<Glycemie> importFichier() {

        logger.debug("importFichier début ----------------");

        List<Glycemie> retour = new ArrayList<>();

        File repertoire = new File(path);
```

```
 7
 8      private static final Logger logger = LogManager.getLogger(LectureFichierGlycemieImpl.class);
 9
10      @Value("${local.path.glycemie}")
11      private String path;
12
13      @Value("${local.path.glycemie.done}")
14      private String done;
15
16      @Override
17      public List<Glycemie> importFichier() {
18
19          logger.debug("importFichier début ----------------");
20
21          List<Glycemie> retour = new ArrayList<>();
22
23          File repertoire = new File(path);
24
25          logger.debug("Vérification que le chemin fourni est bien un répertoire");
26          if (!repertoire.isDirectory()) {
27              logger.error("\"" + repertoire.toString() + "\""
28                      + " n'est pas un répertoire. Arrêt du traitement.");
29
```

```java
    @Override
    public List<Glycemie> importFichier() {

        logger.debug("importFichier début ----------------");

        List<Glycemie> retour = new ArrayList<>();

        File repertoire = new File(path);

        logger.debug("Vérification que le chemin fourni est bien un répertoire");
        if (!repertoire.isDirectory()) {
            logger.error("\"" + repertoire.toString() + "\""
                    + " n'est pas un répertoire. Arrêt du traitement.");

            return retour;
        }

        List<File> fichiersDuRepertoire = Arrays.asList(Objects.requireNonNull(repertoire.listFiles()));

        logger.debug("Vérification de la présence de fichiers à traiter");
        if (CollectionUtils.isEmpty(fichiersDuRepertoire)) {
            logger.info("Aucun fichier à traiter dans"
                    + "\"" + path.toString() + "\".");
```

```
27              logger.error("\"" + repertoire.toString() + "\""
28                      + " n'est pas un répertoire. Arrêt du traitement.");
29
30              return retour;
31          }
32
33          List<File> fichiersDuRepertoire = Arrays.asList(Objects.requireNonNull(repertoire.listFiles()));
34
35          logger.debug("Vérification de la présence de fichiers à traiter");
36          if (CollectionUtils.isEmpty(fichiersDuRepertoire)) {
37              logger.info("Aucun fichier à traiter dans"
38                      + "\"" + path.toString() + "\".");
39
40              return retour;
41          }
42
43          logger.debug("Traitement des fichiers.");
44          for (File fichier:fichiersDuRepertoire) {
45              logger.debug("Traitement de l'élément "
46                      + "\"" + fichier.getName() + "\".");
47
48              retour.addAll(lectureFichier(fichier));
49          }
```

```
        List<File> fichiersDuRepertoire = Arrays.asList(Objects.requireNonNull(repertoire.listFiles()));

        logger.debug("Vérification de la présence de fichiers à traiter");
        if (CollectionUtils.isEmpty(fichiersDuRepertoire)) {
            logger.info("Aucun fichier à traiter dans"
                        + "\"" + path.toString() + "\".");

            return retour;
        }

        logger.debug("Traitement des fichiers.");
        for (File fichier:fichiersDuRepertoire) {
            logger.debug("Traitement de l'élément "
                        + "\"" + fichier.getName() + "\".");

            retour.addAll(lectureFichier(fichier));
        }

        logger.debug("importFichier fin ----------------");
        return retour;
    }

    private List<Glycemie> lectureFichier(File fichier) {
```

```
41          }
42
43          logger.debug("Traitement des fichiers.");
44          for (File fichier:fichiersDuRepertoire) {
45              logger.debug("Traitement de l'élément "
46                      + "\"" + fichier.getName() + "\".");
47
48              retour.addAll(lectureFichier(fichier));
49          }
50
51          logger.debug("importFichier fin ----------------");
52          return retour;
53      }
54
55      private List<Glycemie> lectureFichier(File fichier) {
56
57          logger.debug("lectureFichier début ----------------");
58
59          List<Glycemie> lignesDuFichier = new ArrayList<>();
60
61          logger.debug("Vérification qu'il s'agit bien d'un fichier.");
62
63          if (fichier.isDirectory()) {
```

```
45              logger.debug("Traitement de l'élément "
46                      + "\"" + fichier.getName() + "\".");
47
48              retour.addAll(lectureFichier(fichier));
49          }
50
51          logger.debug("importFichier fin ----------------");
52          return retour;
53      }
54
55      private List<Glycemie> lectureFichier(File fichier) {
56
57          logger.debug("lectureFichier début ----------------");
58
59          List<Glycemie> lignesDuFichier = new ArrayList<>();
60
61          logger.debug("Vérification qu'il s'agit bien d'un fichier.");
62
63          if (fichier.isDirectory()) {
64              logger.info("\"" + fichier.toString() + "\""
65                      + " est un répertoire. Passage à l'élément suivant de la liste.");
66
67              return lignesDuFichier;
```

```
52          return retour;
53      }
54
55      private List<Glycemie> lectureFichier(File fichier) {
56
57          logger.debug("lectureFichier début ----------------");
58
59          List<Glycemie> lignesDuFichier = new ArrayList<>();
60
61          logger.debug("Vérification qu'il s'agit bien d'un fichier.");
62
63          if (fichier.isDirectory()) {
64              logger.info("\"" + fichier.toString() + "\""
65                      + " est un répertoire. Passage à l'élément suivant de la liste.");
66
67              return lignesDuFichier;
68          }
69
70          if (!fichier.isFile()) {
71              logger.error("\"" + fichier.toString() + "\""
72                      + " n'est pas un fichier. Passage à l'élément suivant de la liste.");
73
74              return lignesDuFichier;
```

```
60
61          logger.debug("Vérification qu'il s'agit bien d'un fichier.");
62
63          if (fichier.isDirectory()) {
64              logger.info("\"" + fichier.toString() + "\""
65                      + " est un répertoire. Passage à l'élément suivant de la liste.");
66
67              return lignesDuFichier;
68          }
69
70          if (!fichier.isFile()) {
71              logger.error("\"" + fichier.toString() + "\""
72                      + " n'est pas un fichier. Passage à l'élément suivant de la liste.");
73
74              return lignesDuFichier;
75          }
76
77          try {
78              logger.info("Traitement du fichier " + fichier.toString() + ".");
79              FileInputStream fis = new FileInputStream(fichier);
80
81              InputStreamReader isr = new InputStreamReader(fis, StandardCharsets.UTF_8);
82              BufferedReader br = new BufferedReader(isr);
```

```
            return lignesDuFichier;
        }

        if (!fichier.isFile()) {
            logger.error("\"" + fichier.toString() + "\""
                        + " n'est pas un fichier. Passage à l'élément suivant de la liste.");

            return lignesDuFichier;
        }

        try {
            logger.info("Traitement du fichier " + fichier.toString() + ".");
            FileInputStream fis = new FileInputStream(fichier);

            InputStreamReader isr = new InputStreamReader(fis, StandardCharsets.UTF_8);
            BufferedReader br = new BufferedReader(isr);

            String line;
            int numLigne = 0;

            while ( ( line = br.readLine() ) != null )
            {
                String[] ligne = line.split("\t", -1);
```

```
81              InputStreamReader isr = new InputStreamReader(fis, StandardCharsets.UTF_8);
82              BufferedReader br = new BufferedReader(isr);
83
84              String line;
85              int numLigne = 0;
86
87              while ( ( line = br.readLine() ) != null )
88              {
89                  String[] ligne = line.split("\t", -1);
90                  numLigne++;
91
92                  logger.debug("Ligne non vide, appel à transformeLigne.");
93                  Glycemie record = transformeLigne(ligne, numLigne);
94
95                  if (record != null) {
96                      lignesDuFichier.add(record);
97                      logger.debug("Décodage de la ligne effectué, ajout du record à lignesDuFichier.");
98                  }
99                  else {
100                     logger.error("Erreur de décodage de la ligne " + numLigne + " du fichier " + fichier.toString()
101                     logger.error("Contenu en erreur : ");
102                     logger.error("-" + line + "-");
103                 }
104             }
```

```
90                  numLigne++;
91
92                  logger.debug("Ligne non vide, appel à transformeLigne.");
93                  Glycemie record = transformeLigne(ligne, numLigne);
94
95                  if (record != null) {
96                      lignesDuFichier.add(record);
97                      logger.debug("Décodage de la ligne effectué, ajout du record à lignesDuFichier.");
98                  }
99                  else {
100                     logger.error("Erreur de décodage de la ligne " + numLigne + " du fichier " + fichier.toString()
101                     logger.error("Contenu en erreur : ");
102                     logger.error("-" + line + "-");
103                 }
104             }
105
106             fis.close();
107
108         } catch (IOException e) {
109             logger.error("Une erreur est survenue lors du traitement du fichier "
110                     + fichier.toString() + ".");
111
112             e.printStackTrace();
```

```
                else {
                    logger.error("Erreur de décodage de la ligne " + numLigne + " du fichier " + fichier.toString()
                    logger.error("Contenu en erreur : ");
                    logger.error("-" + line + "-");
                }
            }

            fis.close();

        } catch (IOException e) {
            logger.error("Une erreur est survenue lors du traitement du fichier "
                        + fichier.toString() + ".");

            e.printStackTrace();
        } finally {
            // Déplacement du fichier
            File destination = new File(path + "/" + done);
            boolean existeDone = destination.exists();

            if (!existeDone) {
                logger.debug("Création du répertoire de destination.");
                existeDone = destination.mkdir();
            }
```

```
108         } catch (IOException e) {
109             logger.error("Une erreur est survenue lors du traitement du fichier "
110                     + fichier.toString() + ".");
111
112             e.printStackTrace();
113         } finally {
114             // Déplacement du fichier
115             File destination = new File(path + "/" + done);
116             boolean existeDone = destination.exists();
117
118             if (!existeDone) {
119                 logger.debug("Création du répertoire de destination.");
120                 existeDone = destination.mkdir();
121             }
122
123             if (existeDone) {
124                 boolean move = fichier.renameTo(new File(destination + "/" + fichier.getName()));
125
126                 if (move) {
127                     logger.info("Déplacement du fichier "
128                     + "\"" + fichier.toString() + "\""
129                     + " vers le répertoire de sauvegarde "
130                     + "\"" + destination.toString() + "\""
131                     + " réussi");
```

```
118             if (!existeDone) {
119                 logger.debug("Création du répertoire de destination.");
120                 existeDone = destination.mkdir();
121             }
122 
123             if (existeDone) {
124                 boolean move = fichier.renameTo(new File(destination + "/" + fichier.getName()));
125 
126                 if (move) {
127                     logger.info("Déplacement du fichier "
128                     + "\"" + fichier.toString() + "\""
129                     + " vers le répertoire de sauvegarde "
130                     + "\"" + destination.toString() + "\""
131                     + " réussi");
132                 }
133                 else {
134                     logger.error("Échec du déplacement du fichier "
135                     + "\"" + fichier.toString() + "\""
136                     + " vers le répertoire de sauvegarde "
137                     + "\"" + destination.toString() + "\"" + ".");
138                 }
139             }
140         }
```

```
122
123             if (existeDone) {
124                 boolean move = fichier.renameTo(new File(destination + "/" + fichier.getName()));
125
126                 if (move) {
127                     logger.info("Déplacement du fichier "
128                     + "\"" + fichier.toString() + "\""
129                     + " vers le répertoire de sauvegarde "
130                     + "\"" + destination.toString() + "\""
131                     + " réussi");
132                 }
133                 else {
134                     logger.error("Échec du déplacement du fichier "
135                     + "\"" + fichier.toString() + "\""
136                     + " vers le répertoire de sauvegarde "
137                     + "\"" + destination.toString() + "\"" + ".");
138                 }
139             }
140         }
141         logger.debug("lectureFichier fin ----------------");
142         return lignesDuFichier;
143     }
144 }
```

```
122
123             if (existeDone) {
124                 boolean move = fichier.renameTo(new File(destination + "/" + fichier.getName()));
125
126                 if (move) {
127                     logger.info("Déplacement du fichier "
128                     + "\"" + fichier.toString() + "\""
129                     + " vers le répertoire de sauvegarde "
130                     + "\"" + destination.toString() + "\""
131                     + " réussi");
132                 }
133                 else {
134                     logger.error("Échec du déplacement du fichier "
135                     + "\"" + fichier.toString() + "\""
136                     + " vers le répertoire de sauvegarde "
137                     + "\"" + destination.toString() + "\"" + ".");
138                 }
139             }
140         }
141         logger.debug("lectureFichier fin ----------------");
142         return lignesDuFichier;
143     }
144 }
```

SYNTHÈSE

- DEBUG, VERBOSE en développement 👷
- INFO, WARNING, ERROR en exploitation 🏭
- WARNING si sans impact immédiat ⚠️
- ERROR 🔥 si action nécessaire pour remédiation 🧯
- penser aux constantes magiques du langage
- identifier les objets et traitements métier 🔧

CONCLUSION

Profitons de notre métier pour en apprendre d'autres !


THANK YOU


Questions?

ME CONTACTER

LinkedIn Virginie CASAVECCHIA

Twitter @La_Fee_Dragee

Podcast PunkinDev de Sylvain Coudert

Feedback :

Actoll recrute !