～ColdFusion Summit 2018 Session～

～トラブルシューティング＆CF2018のパフォーマンス～

2018年11月9日

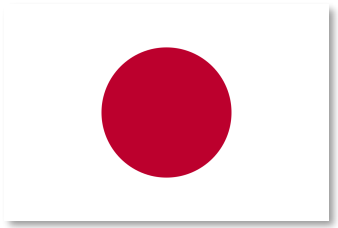
株式会社サムライズ

本日のアジェンダ


WELCOME
LAS VEGAS
NEVADA
Cf
ColdFusion Summit 2018 情報
・トラブルシューティング
・ColdFusion 2018 の
パフォーマンス

ColdFusion Summit 2018 トラブルシューティング

ColdFusion Summit 2018の1日セッションで紹介されたのトラブルについてご紹介します。

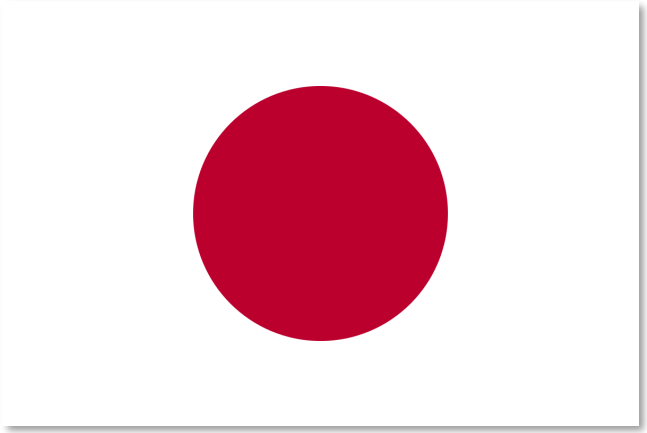


≒

不具合、障害情報が多そう

開発者が多いので回避方法も多いのでは？

ColdFusion Summit 2018 トラブルシューティング



÷

過去のCF Day 技術セッション資料リンク(2014-2017)

https://www.samuraiz.co.jp/event/report/cfday2017/img/CFDay2017_session2.pdf
https://www.samuraiz.co.jp/event/report/cfday2016/img/CFDay2016_session2.pdf
https://www.samuraiz.co.jp/event/report/cfday2015/img/session2_suptopics.pdf
https://www.samuraiz.co.jp/event/report/cfday2014/img/ColdFusionDay2014_02.pdf

ColdFusion Associate
ColdFusion開発会社・パッケージベンダーが集結

http://cfassociates.samuraiz.co.jp/

ColdFusion Summit 2018 トラブルシューティング

メモリ			
要因	知識	確認	対応策
heapサイズの不足	Javaのメモリの構造 Javaの引数の情報	・ヒープの使用量をメトリクスログや監視ツールで確認 ・GCの頻度・所要時間を確認	・heapサイズを調整する
MetaSpaceの不足		・cfclasses内のファイル数が多くなっていないかの確認	・MetaSpaceを増やす ・定期的にclassファイルを削除する
日本の情報			
【P26参照】https://www.samuraiz.co.jp/event/report/cfday2017/img/CFDay2017_session2.pdf			

ハング			
要因	知識	確認	対応策
キューが埋まっている	・cfstat、メトリクスログの知識 ・タイムアウトの設定 ・スタックトレースの知識	・cfstat、メトリクスログ、スナップショットでスレッドの状態を確認、または監視ツールを使用する（RMT等）	・メモリを増やす（不足している場合） ・シングルスレッドのタグ・関数を見直す ・タイムアウトを設定する
シングルスレッドのタグ、関数を使用している			
外部連携での応答待ち			
日本の情報			
https://www.samuraiz.co.jp/event/report/cfday2016/img/CFDay2016_session2.pdf 【P15~】https://www.samuraiz.co.jp/event/report/cfday2014/img/ColdFusionDay2014_02.pdf http://cfassociates.samuraiz.co.jp/index.cfm/faq/cftech/cf10-cfhttp-response/ http://cfassociates.samuraiz.co.jp/index.cfm/faq/cftech/cf-security-egd/			
その他リンク			
https://www.carehart.org/blog/client/index.cfm/2010/10/15/lies_damned_lies_and_CF_timeouts https://www.carehart.org/blog/client/index.cfm/2012/5/28/cf_image_resizing_problem_and_solution https://www.carehart.org/blog/client/index.cfm/2012/5/28/cf_image_resizing_problem_and_solution			

ColdFusion Summit 2018 トラブルシューティング

CPU			
要因	知識	確認	対応策
CF以外のものでCPUが高くなっている (システムメモリの使用量の問題、ディスクのスペース不足)	タスクマネージャやOS監視ツールの知識	・CPUが高くなっているアプリケーションを確認 ・ディスクの空き容量の確認	・余分なキャッシュを削除する ・ログファイルのローテーションを行うことや設定を変更する
アンチウイルスソフト	ー	・不要なフォルダに対してチェックしていないかの確認 ・古いプログラムになっていないかの確認	・チェック対象を変更する ・プログラムを更新する
レジストリ内で処理中のclient変数	レジストリの簡単な知識	・Client変数の使用の有無と管理画面の設定を確認する	・Client変数のレジストリ設定を変更する ・レジストリファイルからクライアント変数を削除する
画像のリサイズ処理数を使用している	ー	・プログラムを確認する	・不要な処理があれば変更する ・サーバを別にする

ColdFusion Summit 2018 トラブルシューティング

CFのアップデート			
要因	知識	確認	対応策
権限のあるユーザーで実行していない	管理者権限の起動方法	・アップデートのインストールログでSuccess、Warning、Errorを確認する	・アップデーターを一旦アンインストールしてから再度アップデートする。 ・アップデートをコマンドで実行する(権限に注意) ・ウイルスソフトを一旦無効にしてから実行する
サービスの停止/起動に時間が掛かり正常なアップデート処理が行われなかった。	イベントログの知識	・イベントログにColdFusionの停止・起動エラーが無いかを確認する	・再度サービス起動して動作するかを確認する ・OSを再起動し余分なアプリケーションが起動していない状態で、アップデーターを一旦アンインストールする。その後、再度アップデートする。
コネクタの再設定をしていなかった	コネクタの設定方法	・ColdFusionのアップデート情報を確認して、コネクタの更新があるかを確認する	・コネクタの削除、追加を行う
日本の情報			
【P18アップデータの実行】https://www.samuraiz.co.jp/event/report/cfday2015/img/session2_suptopics.pdf 【P28起動停止のタイムアウト】https://www.samuraiz.co.jp/event/report/cfday2016/img/CFDay2016_session2.pdf 【P9コネクタの更新】https://www.samuraiz.co.jp/event/report/cfday2014/img/ColdFusionDay2014_02.pdf			

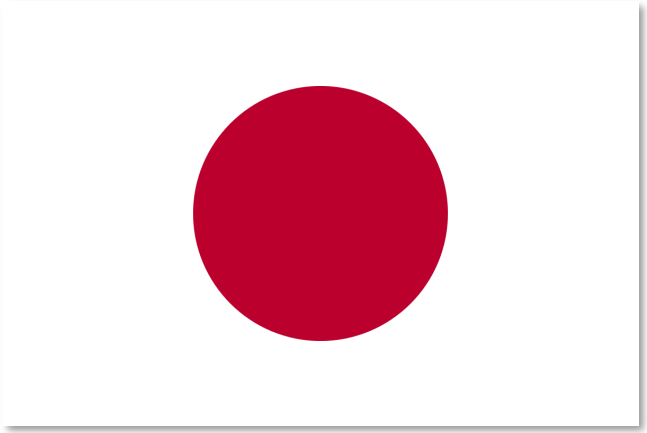
ColdFusion Summit 2018 トラブルシューティング

JVM（JDK）			
要因	知識	確認	対応策
CFHTTPの接続が失敗する	簡単なSSLの知識	・接続先でSSL3.0やTLS1.0が無効になっていないかを確認す ・Javaのバージョンを確認する	・JDKのバージョンを上げる ・ColdFusionをマイグレーションする ・JVMの引数にTLSに関する設定を追加する -Dhttps.protocols=TLSv1.2
起動しない	JVM起動パラメータの知識	・jvm.configを確認する	・ヒープ、metaspaceのメモリの設定値を正しい値に変更する ・JVMのパスが正しいかを確認し、修正する ・JVM引数の指定に誤りがあれば修正する
	—	JDKを1.7から1.8に変えていないかを確認する	ColdFusion 11の場合tools.jarを、使用するJDKに含まれているjarに置き換える。
日本の情報			
【リビジョンアップ】http://cfassociates.samuraiz.co.jp/index.cfm/faq/administrator/coldfusion-jvm/ 【Java8からJava7へ戻す場合】http://cfassociates.samuraiz.co.jp/index.cfm/faq/coldfusion11/cf11-back-to-java7/ 【tools.jarについて】http://cfassociates.samuraiz.co.jp/index.cfm/faq/coldfusion11/cf11-update-3-java8/ 【P.63~】https://www.samuraiz.co.jp/coldfusion/upgrade/img/ColdFusion_tips_2015.pdf			

ColdFusion Summit 2018 トラブルシューティング

CF/Webサーバーのチューニング			
要因	知識	確認	対応策
マルチサイトで応答待ちが発生する	コネクタの設定	マルチサイトの設定になっていないかを確認する。（デフォルトはシングルサイトに適した設定になっているのでマルチサイトの時はチューニングが必要）	・CF2018(PMT)自動チューニング機能を使用する ・max_reuse_connectionをconnection_pool_size÷サイト数にする

ColdFusion Summit 2018 トラブルシューティング



÷

過去のCF Day 技術セッション資料リンク(2014-2017)

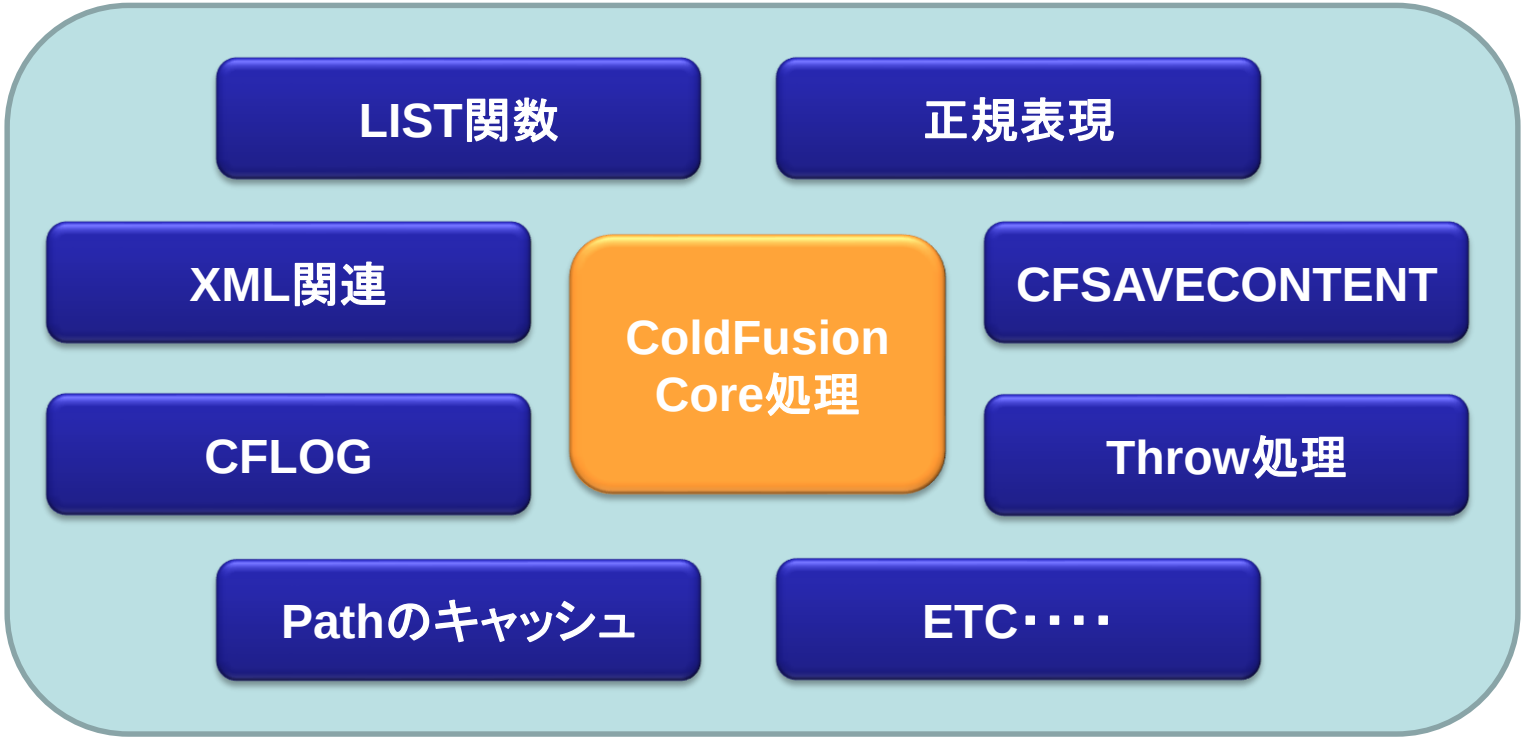
https://www.samuraiz.co.jp/event/report/cfday2017/img/CFDay2017_session2.pdf
https://www.samuraiz.co.jp/event/report/cfday2016/img/CFDay2016_session2.pdf
https://www.samuraiz.co.jp/event/report/cfday2015/img/session2_suptopics.pdf
https://www.samuraiz.co.jp/event/report/cfday2014/img/ColdFusionDay2014_02.pdf

ColdFusion Associate
ColdFusion開発会社・パッケージベンダーが集結

http://cfassociates.samuraiz.co.jp/

ColdFusion 2018 のパフォーマンス

ColdFusion 2018ではColdFusion 2016に比べさらにスピードアップしました。このセッションではアーキテクチャ変更やその効果についてご紹介します。

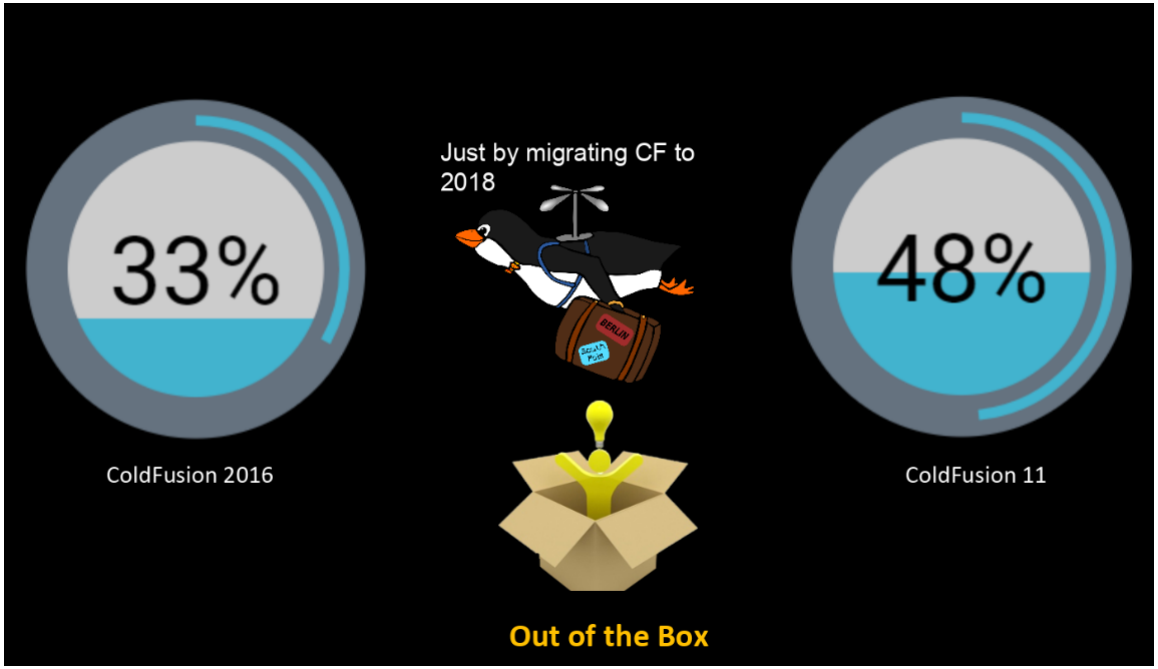


どれくらい改善するの？

プログラム次第のところはありますが、Adobeでは以下のように改善すると告知されています。

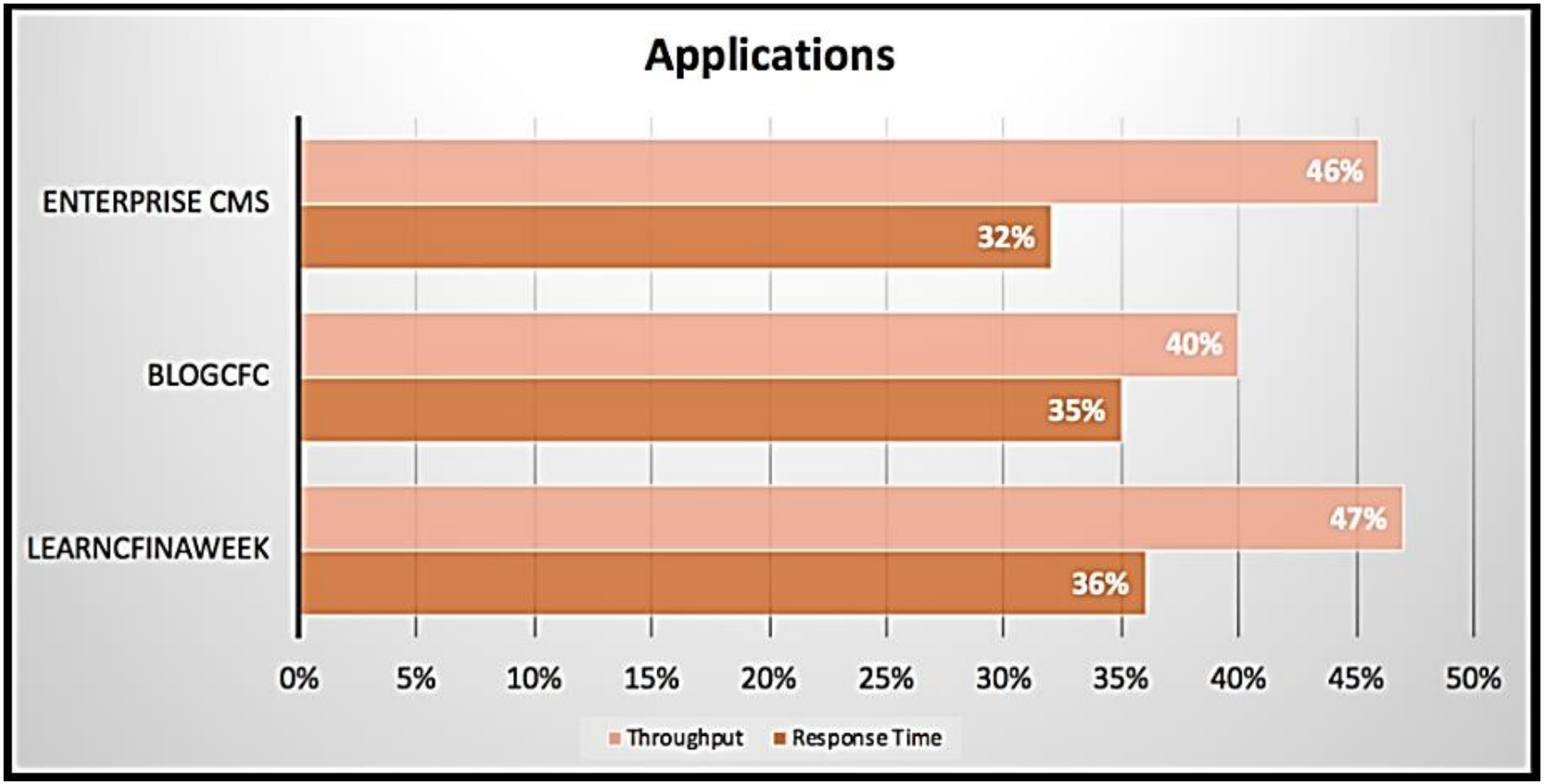
ColdFusion 2016からColdFusion 2018は33%改善

ColdFusion 11からColdFusion 2018は48%改善



どれくらい改善するの？

改善例:



List関連例（ListContains）

これまで”Lasvegas”を検索する場合の方法

Tokyo,Lasvegas,Bangkok,NewYork

Lasvegas　　Lasvegas

CF2018で”Lasvegas”を検索する場合の方法

Tokyo,Lasvegas,Bangkok,NewYork

Lasvegas　　Lasvegas

1文字で検索

List関連の改善

処理アーキテクチャ変更等の最適化の結果、List関数は以下のようにスピードアップされています。

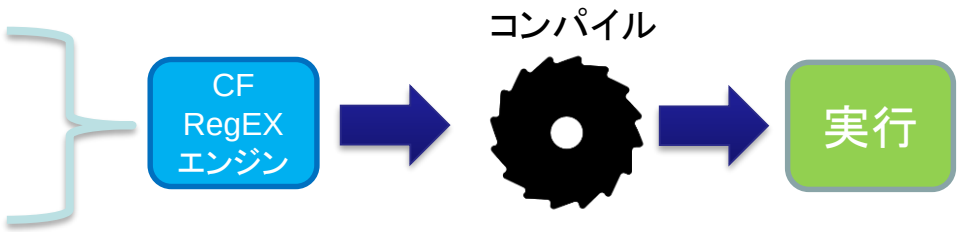
関数	改善値%
LISTPREPEND	30.57%
LISTFIND	42.71%
LISTAPPEND	45.44%
LISTFINDNOCASE	50.51%
LISTCONTAINS	57.69%
LISTLEN	91.75%
LISTINSERTAT	93.25%
LISTDELETEAT	93.25%
LISTSETAT	93.96%
LISTGETAT	96.44%
LISTLAST	99.94%

正規表現の改善

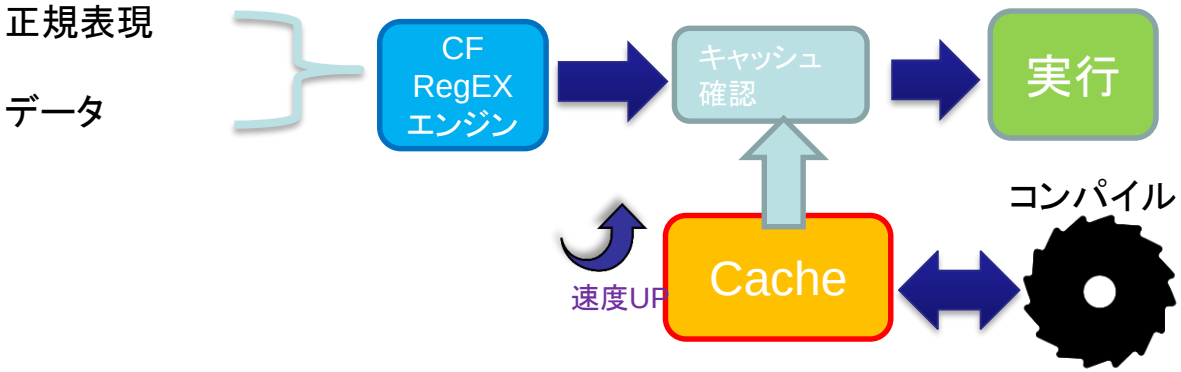
正規表現を使用する場合、Regular Expressionを毎回コンパイルして使用していました。

正規表現:
<([A-z][A-Z0-9]*¥b[^>]*>(.*?)<¥/¥1>

データ:
<TAG>one<TAG>two</TAG>one</TAG>

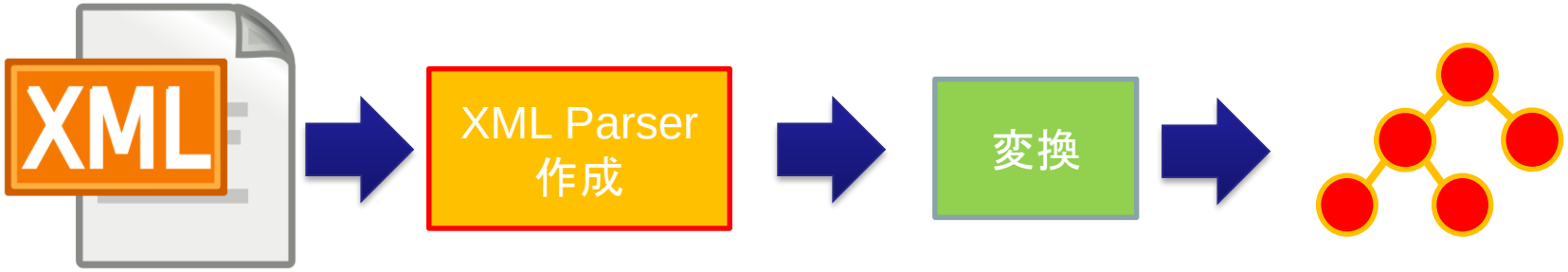


CF2018ではコンパイルデータをキャッシュし、再利用することでスピードアップすることができるようになりました。

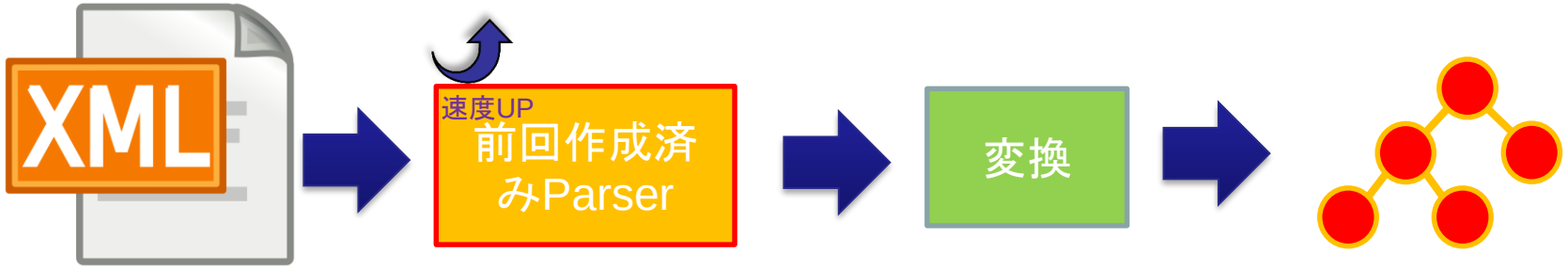


XML関連の改善

XMLを使用する際毎回XML parserを作成していました



CF2018ではこのparserをプールし再利用することで2回目以降の処理が高速化されるようになりました。



XML関連の改善

処理アーキテクチャ変更の結果、XML関連処理は以下のように改善されています。

速度UP 関数	改善値%
XML Validate(validator)	67.39%
XML Validate	76.95%
IsXML	83.21%
WDDX Deserialize	86.36%
XML Parse(Validator)	90.09%
IsXML Negative	92.26%
XML Parse	93.87%

savecontentの改善

Xmlの作成等多くのケースに使用されているsavecontentで速度が向上しました。

```
<cfloop from="1" to="#url.loopcount#" index="y">
        <cfsavecontent variable="myContent">
                <cfset binaryString=
toBinary(toBase64("foobar"))>
                <cfset binecode=
BinaryEncode(binaryString,"UU")>
        </cfsavecontent>
        <cfsavecontent variable="myContent">
                <cfset inputString = "barfoo">
                <cfset hashString =
hash(inputString,"MD5","UTF-8")>
        </cfsavecontent>
</cfloop>
```

	実行結果 (改善率)
CF2018 cfsavecontent有	**150ms(93%)**
CF2018 cfsavecontent無	90ms
CF2016 cfsavecontent有	**2200ms**
CF2016 cfsavecontent無	100ms

速度UP

LOGの改善

エラー出力や、システムのデバック出力に多く使用される
ログの出力がファイルI/O操作の最適化によりスピードアップされています。

cflog	実行結果
CF2018	120ms(**94%**) 速度UP
CF2016	2002ms

```
for(i=1; i LTE url.samplesCount; i++)
{
        timeStart = getTickCount();
        for(loopCounter=1; loopCounter LTE url.loopCount; loopCounter++)
        {
                cflog(file ="myAppLig1", application="no", text="test no app log.");
                cflog(file ="myAppLig2", application="no", text="test app log.");
        }
        request.testTimeSamples[i] = getTickCount()-timeStart;
        writeOutput("time elapsed:" & request.testTimeSamples[i] & "<br>");
}
```

Throw処理の改善

ThreadのStacktraceはとても重い物でしたがCF2018では
ランタイムの実行ブロックの一部をコンパイル時に行うことで、スピードアップされています。

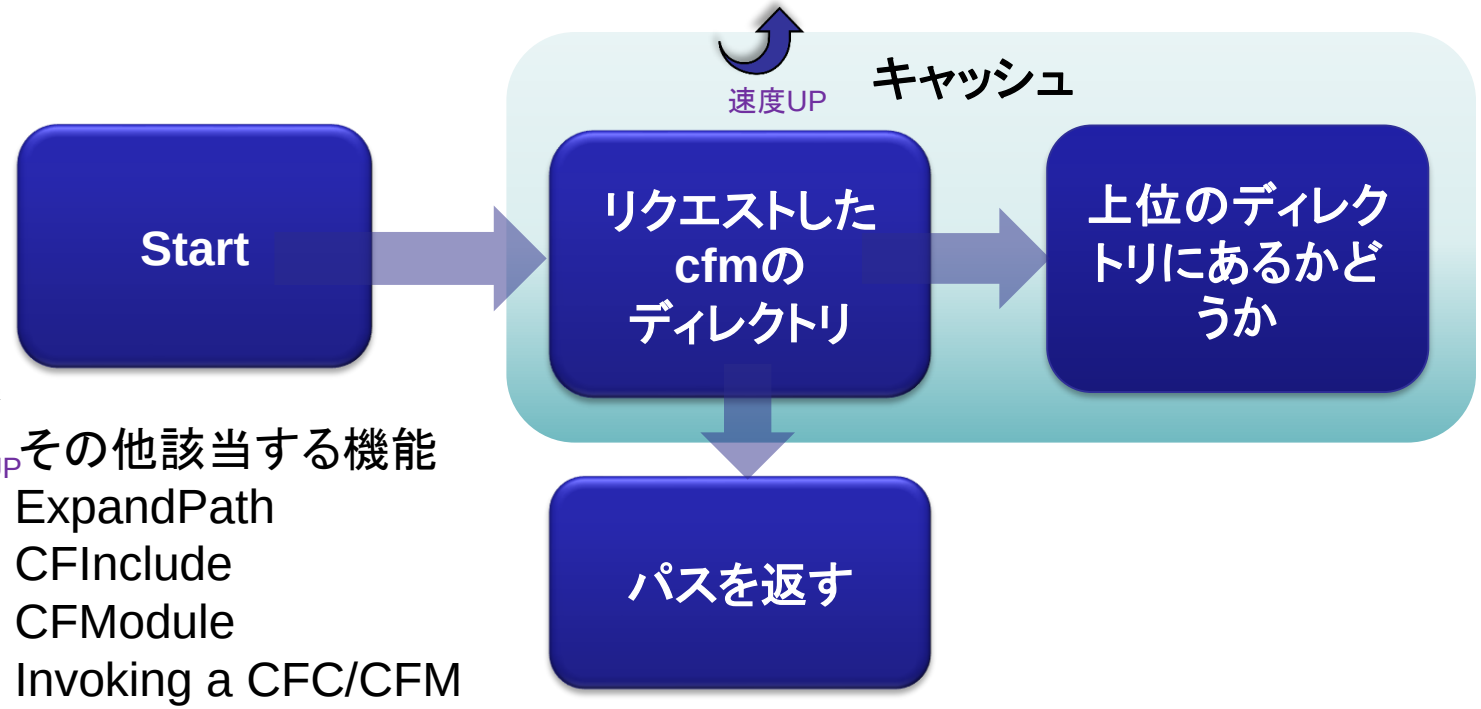
```
for(loopCounter=1; loopCounter LTE url.loopCount; loopCounter++)
{
    try{
        throw(message="Oops",detail="xyz");
    }catch(any e){
        writeOutput("Error:" & e.message);
    }finally{
        writeOutput("I run even if no error");
    }
}
```

throw	実行結果
CF2018	931ms(**87%**)
CF2016	7395ms

速度UP

Pathのキャッシュの改善

Application.cfcはリクエストしたcfmと同じフォルダにないと上位のフォルダに検索していきます。これは、毎回おこなわれているため、パフォーマンスに影響します。
これはCache web Server pathsという項目を有効にすることでキャッシュされるようになり、改善します。
※ColdFusion 2018よりマルチサイトでも使用できるようになりました。



速度UP その他該当する機能

- ExpandPath
- CFInclude
- CFModule
- Invoking a CFC/CFM

Pathのキャッシュの改善

ExpandPathをループした場合の例:

ExpandPath	実行結果
CF2018 パスキャッシュ有	**12ms(99%)**  速度UP
CF2018 パスキャッシュ無	800ms
CF2016 パスキャッシュ無	**800ms**
CF2016 パスキャッシュ有	800ms

試しに・・・・

簡易アプリケーションの結果をご紹介します。

CF2016

デバック出力

replaceの回数：45194
StructFindの回数：125
listGetAtの回数：26198
RESTリクエスト数：28

result.csv ダウンロード dataset.csv Identifier名称抜粋 ダウンロード

Debugging Information

ColdFusion Server デベロッパー	2016,0,07,311392
Template	/gd2/viewall.cfm
Time Stamp	15-Oct-18 09:20 PM
Locale	Japanese
User Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/69.0.3497.100 Safari/537.36
Remote IP	0:0:0:0:0:0:0:1
Host Name	0:0:0:0:0:0:0:1

Execution Time

Total Time	Avg Time	Cou	Template
19608 ms	19608 ms	1	C:/ColdFusion2016/cfusion/wwwroot/gd2/viewall.cfm
			STARTUP, PARSING, COMPILING, LOADING, & SHUTDOWN
20178 ms			*TOTAL EXECUTION TIME*

red = over 250 ms average execution time

試しに・・・・

簡易アプリケーションの結果をご紹介します。

CF2018

デバック出力

---

replaceの回数：45194
StructFindの回数：125
listGetAtの回数：26198
RESTリクエスト数：28

---

result.csv ダウンロード dataset.csv Identifier名称抜粋 ダウンロード

Debugging Information

ColdFusion Server Evaluation	2018,0,0,310739
Template	/gd2/viewall.cfm
Time Stamp	15-Oct-18 09:20 PM
Locale	Japanese
User Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/64.0.3282.140 Safari/537.36 Edge/17.17134
Remote IP	0:0:0:0:0:0:0:1
Host Name	0:0:0:0:0:0:0:1

Execution Time

Total Time	Avg Time	Cour	Template
10685 ms	10685 ms	1	C:/ColdFusion2018/cfusion/wwwroot/gd2/viewall.cfm
			STARTUP, PARSING, COMPILING, LOADING, & SHUTDOWN
11291 ms			*TOTAL EXECUTION TIME*

red = over 250 ms average execution time

CF2018 : 10685ms　　CF2016 : 19608ms

デバック出力

replaceの回数：45194
StructFindの回数：125
listGetAtの回数：26198
RESTリクエスト数：28

result.csv ダウンロード dataset.csv

Debugging Information

ColdFusion Server Evaluation	2018,0,0,310739
Template	/gd2/viewall.cfm
Time Stamp	15-Oct-18 09:20 PM
Locale	Japanese
User Agent	Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebk
Remote IP	0:0:0:0:0:0:0:1
Host Name	0:0:0:0:0:0:0:1

Execution Time

Total Time	Avg Time	Count	Template
10685 ms	10685 ms	1	C:/ColdFusion2018/cfusion/wwwroot/gd2/viewall.cfm
606 ms			*STARTUP, PARSING, COMPILING, LOADING, & SHU*
11291 ms			*TOTAL EXECUTION TIME*

red = over 250 ms average execution time

45%
改善!!

デバック出力

replaceの回数：45194
StructFindの回数：125
の回数：26198
28

dataset.csv Identifier名称抜粋 ダウンロード

ッパー 2016,0,07,311392
/gd2/viewall.cfm
15-Oct-18 09:20 PM
Locale　Japanese
User Agent　Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537
Remote IP　0:0:0:0:0:0:0:1
Host Name　0:0:0:0:0:0:0:1

Execution Time

Total Time	Avg Time	Count	Template
19608 ms	19608 ms	1	C:/ColdFusion2016/cfusion/wwwroot/gd2/viewall.cfm
570 ms			*STARTUP, PARSING, COMPILING, LOADING, & SHUTDOWN*
20178 ms			*TOTAL EXECUTION TIME*

red = over 250 ms average execution time

まとめ

今回ご紹介した機能や関数は、一般的な業務に多く使用されるようなものばかりです。また、これらは、現在のシステムのプログラムを変更することなく、恩恵を受けるコアな処理です。

既存のシステムで ColdFusion2018の速度を 体感してみませんか？

参考：パフォーマンスのホワイトペーパー
https://wwwimages2.adobe.com/content/dam/acom/en/products/coldfusion/pdfs/CF2018_Performance.pdf