# Nextra
# AppMinder ユーザガイド

---

Version 5
2nd Edition

# 目 次

**第 1 章 はじめに ..... 2**

本書の利用方法 ..... 2
表記規則 ..... 3

**第 2 章 AppMinder の特徴と機能 ..... 5**

AppMinder の概要 ..... 5
多重ブローカ階層 ..... 9
セッションセキュリティ ..... 10
ネーミング・サービスのクリーンアップ ..... 10

**第 3 章 AppMinder の起動 ..... 13**

起動の前に ..... 13
AppMinder モニタの起動 ..... 14
AppMinder エージェントの起動 ..... 16
ビューアの起動 ..... 21

**第 4 章 AppMinder ビューアの使用 ..... 22**

モニタとの接続 ..... 22
設定の検索 ..... 23
設定の更新 ..... 27
プロセスのステータスシンボル ..... 31

**第 5 章 AppMinder のコマンドラインインタフェースの使用 ..... 33**

概要 ..... 33
サービスの起動と停止 ..... 34
AppMinder プロセスの停止 ..... 36

**第 6 章 チュートリアル ..... 37**

環境ファイルと初期化ファイルの作成 ..... 38
AppMinder コンポーネントの起動 ..... 40
設定の作成 ..... 43
設定をアクティブにする ..... 48
シャットダウン ..... 51
便利な機能 ..... 52

# 第 1 章 はじめに

この章では、『AppMinder ユーザガイド』の使用方法、対象読者、説明項目の概要、表記規則について説明します。

## 本書の利用方法

本書『AppMinder ユーザガイド』では、分散アプリケーションの構築や管理を行う際に使用できる、より複雑な機能を紹介するために、重要な項目についてステップを追って説明します。この後を読んで、対象読者に該当し、本書が必要に応じた適切なマニュアルであることを確認してください。

### 対象読者

本書は、3 層分散アプリケーションを監視するために AppMinder を使用する管理者を対象としています。

### 前提知識

本書は、読者がクライアント／サーバコンピューティングと 2 層分散アーキテクチャの制限について基本的に理解していることを前提にしています。本書を読む前に、『はじめにお読みください』、『サーバ開発者ガイド』、『運用／設定ガイド』、『セキュリティガイド』をお読みください。

### 本書の使用方法

『AppMinder ユーザガイド』は、AppMinder の動作について必要となるすべての情報を提供します。AppMinder のユーザガイドおよびリファレンスとしてお使いください。

## 本書の内容

本書では、次の項目について説明しています。

- AppMinder の機能と能力
- AppMinder の各コンポーネントの起動
- AppMinder のグラフィカル・ユーザ・インタフェース(Viewer)による設定管理
- 設定管理のための AppMinder コマンドラインインタフェースの使用

# 表記規則

## 文中の表記規則

本書で使用する規則を理解しておくと、ユーティリティの使用方法などを容易に理解できます。

| 形式 | 説明 | 例 |
|---|---|---|
| terminal | OS やサードパーティのユーティリティ、ファイル名または変数の定数値を示します。 | telnet<br><br>cust.def |
| *sub-text* | ユーザが指定する必要がある値を示します。 | *server*_c.pl<br><br>-e *environment_file* |
| **bold** | 本文中では Nextra ユーティリティを示します。サンプル中では、強調される部分を示します。 | **rpcperl**<br><br>**RPCMake** |
| [brackets] | \| がない場合は、オプションテキストを示します。\| がある場合は、いずれか 1 つを選択することを示します。 | [-d *def_file*]<br><br>[NONE \| ERROR \| WARN \| DEBUG] |

次の形式で区別されているパラグラフは、コード例です。

```
#include <stdio.h>

main() {
        int i;
        printf("The number is %d\n",i);
}
```

## 本書で使用するシンボル

本書では、次のようなシンボルを使用しています。


**警告メッセージ**

このシンボルに続くメッセージに、特別な注意を払う必要があることを示しています。このメッセージには重要な情報が含まれており、この情報を正しく理解してから先に進んでください。

**ヒントメッセージ**

このシンボルに続く本文は、必須ではありませんが状況に応じて役立つ手順であることを示しています。

**オプションメッセージ**

このシンボルに続く本文はオプションであることを示しています。内容は、追加機能または代替手法の概要、ある概念を理解するために役立つプロセスステップの詳細などです。

**デバッグ方法**

このシンボルに続く本文は、プロジェクトの現在のステップをデバッグする手順が含まれていることを示しています。この方法はあくまで参考であり、別の有効なデバッグ方法の使用を妨げるものではありません。

# 第 2 章 AppMinder の特徴と機能

この章では、**AppMinder** の特徴を紹介し、**AppMinder** の機能の詳細について説明します。機能を理解した後、「AppMinder の起動」の章に進んでください。

この章を読む前に、『サーバ開発者ガイド』で説明されている内容をご理解ください。**AppMinder** の主な機能は、分散アプリケーションを起動し、監視することなので、多重ブローカ階層を含む分散アプリケーションの構造について理解しておく必要があります。

## AppMinder の概要

**AppMinder** ユーティリティは、Nextra オープン開発環境内で以下のネットワーク管理サービスを提供します。

- サービス(サーバ、ブローカ)を起動する。
- サーバとブローカが実行されていることを確認するために、定期的にサービスをチェックする。
- ブローカがダウンした場合、**AppMinder** はブローカを再起動し、そのブローカに登録されていたすべてのサービスを再登録する。
- サーバがダウンした場合、**AppMinder** はそのサーバを再起動する。

**AppMinder** の主な機能は、分散アプリケーションの起動と監視です。**AppMinder** は、クライアント、サーバおよびブローカが通信を行うのには必要ではありませんが、分散アプリケーションの堅牢性を高めます。

### AppMinder の各コンポーネントの概要

**AppMinder** の使い方を理解するために、**AppMinder** の各コンポーネントについて理解する必要があります。**AppMinder** は、3 層分散アプリケーションであり、図 2.1 に網掛けで示した 3 つのコンポーネントから成ります。

このモデルでは、グラフィカル・ユーザ・インタフェース(GUI)であるビューア(AmViewer)を設定の編集・表示に使用しています。「設定」とは、単一のアプリケーションを構成するサーバとブローカの階層リストのことです。「アクティブ設定」とは、現在 **AppMinder** が管理している設定のことです。

モニタ(Monitor)は、ビューアから要求を受け取り、エージェントに転送し、サーバ状態情報をビューアに返します。

エージェント(Agent)は、モニタから受け取った要求を実行してサーバの開始と停止を行います。コンポーネント間での通信にはリモート・プロシージャ・コール(RPC)を使用します。

次節では、これらの 3 つのコンポーネントについて詳しく説明します。

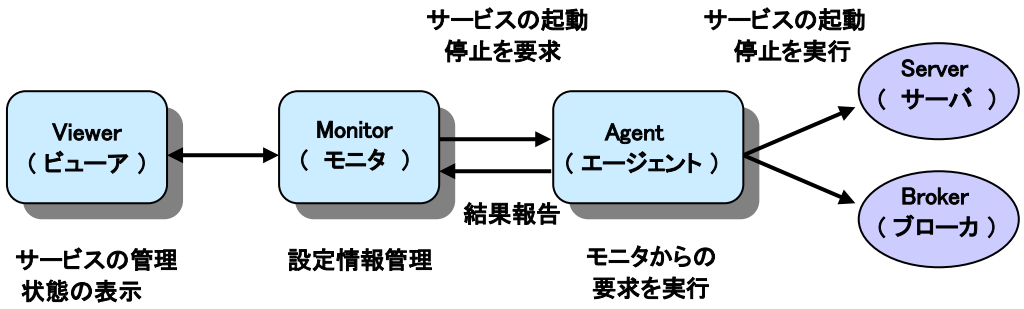


図 2.1： AppMinder のコンポーネントビューア

## ビューア

**AppMinder** のビューアは、どのプロセスが動作中なのかを表示し、アプリケーションの管理を容易にします。ビューアは Windows プラットフォームで動作し、設定の編集とアクティブ時の状態監視を可能にします。

ビューアは Nextra RPC を使ってモニタと通信します。

## モニタ

モニタは、複数の設定を同時に管理し、スタンバイ時に編集を可能にします。

モニタは、すべてのサービスに関する情報をメモリ内に格納します。

## エージェント

モニタは、エージェントを使って、任意のローカルマシンまたはリモートマシンの上でサーバの起動、停止、および再起動を実行します。エージェントとは、バックグラウンドで絶えず動作しながら、モニタプロセスからの要求を待つプロセスのことです。


**エージェントのローカルファイル（appmagt.cdb）**

あるホスト上におけるサービスの起動または停止を指示すると、モニタは、そのホストで動作しているエージェントに要求を送ります。そのあと、エージェントは、この要求を実行します。各エージェントは、エージェントのホスト上で動作しているすべてのサービスに関する情報が格納されたローカルファイルを保守します。エージェントのローカルファイル（appmagt.cdb）は、絶対に手動で変更しないでください。

また、-recover のオプションを付けて起動する場合は、このローカルファイルが必要となりますのでご注意ください。

## 機能のまとめ

AppMinder の機能について簡単に説明します。

### 内部セキュリティ

・ セッションセキュリティ

モニタとエージェントは、常に、暗号化 RPC を使って通信を行います。

### 構成ファイル

AppMinder のビューアを使って分散アプリケーション階層構造を対話式に作成、デバッグすることができます。作成が完了したら、階層設定を設定ファイルとして保存することができます。(ファイル名を指定します。) 保存後は、この設定ファイルを好きなときにロードし、サーバとブローカの設定を希望どおりに、ただちに簡単に再生することができます。AppMinder では、設定をパスワードでロックして、他のユーザが設定ファイルを読むことしかできないようにすることもできます。

### ビューアを使用しない場合

**AppMinder** は、3 層分散アプリケーションであり、プレゼンテーションからビジネスロジックが分離されているため、設定動作が完了した後にビューアを終了しても、モニタとエージェントにバックグラウンドで設定内容を管理させることができます。

## 複数のブローカ階層

設定には、すべてのサーバの位置情報を登録するマスタブローカが常に 1 つ必要です。マスタブローカは、「サブブローカ」と呼ばれる他のブローカの上位になることもできます。各サブブローカは、自分自身が管理するサーバと、自分の下位に位置するブローカの位置情報を保持します。

**AppMinder** における複数ブローカの監視機能の詳細については、「多重ブローカ階層」を参照してください。

## エラーチェック

**AppMinder** には数種類のログ機能があります。モニタの起動時にログファイルのファイル名を指定し、モニタ自体の動作とエラーについての情報を、そこに書き込ませることができます。同様に、エージェントが自分の動作とエラーを記録するためのログファイルを指定することができます。また、モニタとエージェントについてのログファイルを指定し、2 つのコンポーネント間の RPC 通信に関する情報のログをとることもできます。

## 呼び出し可能 API

現在のバージョンではサポートされていません。

## オーファンサーバの復旧

動作を停止するとき、エージェントは、自分が管理しているすべてのサーバも停止します。ただし、状況によって、エージェントの停止後にも動作が続行している「オーファンサーバ」が存在することがあります。**AppMinder** では、復旧オプションを使って、再起動時にオーファンサーバを探すようにエージェントに指示し、エージェントを起動することができます。エージェントがオーファンサーバを見つけた場合、そのオーファンサーバに対して管理を続行するので、サーバプロセスを新たに起動することが不要になります。

### AppMinder の微調整

モニタとエージェントには、それら自体の動作を微調整することのできる多くのオプションがあります。これらのオプションは、モニタとエージェントが起動時に使用する初期化ファイルに設定することができます。

### サービスのスケジューリング

AppMinder では、サービスプロセスとサーバプロセスに対して、日単位、週単位、月単位で起動停止時刻をスケジュールすることができます。

## 多重ブローカ階層

AppMinder は、多重ブローカ階層もサポートします。多重ブローカ階層は常に、他のサービス、つまり、サーバまたはブローカとして定義される「サービス」を登録するマスタブローカが 1 つ必要になります。マスタブローカ以外のブローカはすべて「サブブローカ」と呼ばれます。個々のサブブローカは、自分自身のサービス情報を、順番に上位のブローカに伝達します。この階層は、マスタブローカをルートとするサービスのツリー構造と似ています。(このツリーは、家系図と同様、上から下に向かう点に注意してください。)

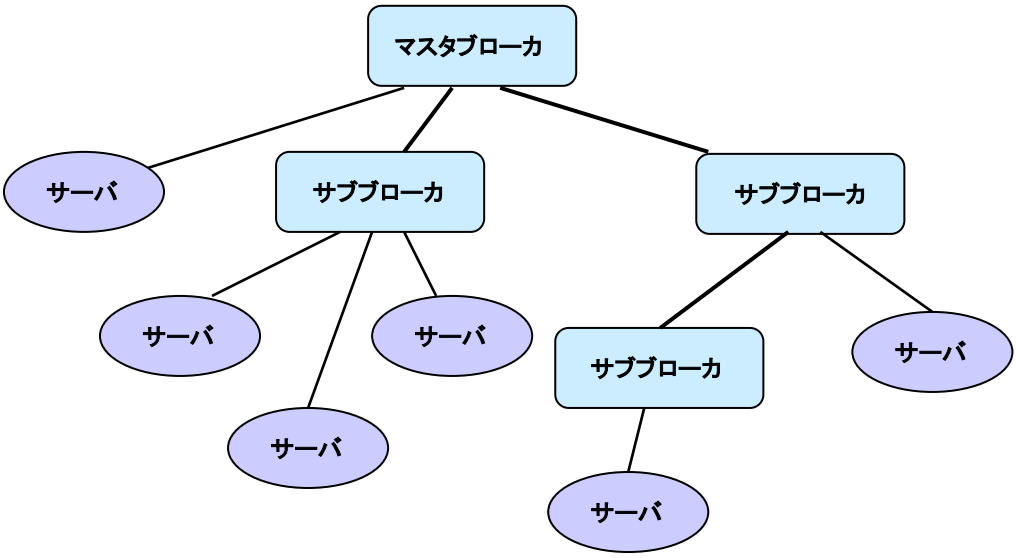


図 2.2： 多重ブローカ階層

### 多重ブローカ構造の長所

多重ブローカ構造には、2 つの大きな長所があります。

### 負荷を最小にする

1 つ以上のブローカが、アプリケーションサービスを管理できるので、1 つのブローカにかかる負荷が最小になります。これはブローカの仕事をツリーの枝に分散することになります。

次のような例を挙げられます。会社 A は、21 のサーバから成るアプリケーションを持っています。アプリケーション管理者は、4 つのサブブローカとマスタブローカを起動し、サーバの負荷を分散しています。4 つのサーバをサブブローカ 1 で、4 つをサブブローカ 2 で、残りの 6 つと 7 つをサブブローカ 3 と 4 でそれぞれ管理し、各サーバの位置情報を保持し、クライアントのリクエストに答えます。

### 堅牢性

2 番目の長所は堅牢性です。多重ブローカ構造では、複数サブブローカの中の 1 つが何らかの理由で利用できなくなった場合にも、サーバは他のサブブローカにアクセスできます。

## セッションセキュリティ

モニタとエージェントは、互いの間で通信するために常に暗号化 RPC を使用します。モニタとエージェントは、起動時に、ともにパスワードを指定する必要があります。2 つのパスワードは一致しなければなりません。モニタとエージェントは、このパスワードを使って互いに RPC 通信を暗号化します。

## ネーミング・サービスのクリーンアップ

サービスの起動・停止に加えて、**AppMinder** エージェントは、そのサービスの状態を絶えずチェックしています。そして、それ自体のサービスリストを、ブローカが保持するリストと比較します。この節では、エージェントがサービスをどう管理するかについて説明します。

### エージェントの起動

エージェントは起動時に、与えられたコマンドラインの引き数、および初期化ファイルに基づき、自己初期化を行います。そのあと、エージェントは、設定 DB にリストされているサービスを起動しようと試みます。設定 DB が存在しない場合は、エージェントは、空の設定 DB をディスクに書き込みます。

## 管理サイクル

起動後、エージェントは管理サイクルを実行します。各サイクルにおいて、エージェントは、リスト内の各サービスを 1 回チェックし、サービスの状態に対応する管理動作を実行します。

管理サイクルでエージェントがどう動作するかについて、以下で説明します。

1. エージェントは、リストの先頭にあるサービスの状態をチェックします。

2. サービスが PING 状態にある場合、エージェントは、そのサービスに ping しようと試みます。ping に失敗すると、エージェントはサービスの状態を FAILED に変更し、起動試行カウンタをインクリメントした後、モニタに状態を更新するよう要求を送ります。

3. サービスが STARTED 状態にある場合、エージェントは、サービスが起動してからの経過時間をチェックします。

   遅延時間が満了している場合、管理サイクルの次のループでサービスを ping することがわかるように、エージェントはサービスの状態を VERIFYING に変更し、モニタに状態を更新するよう要求を送ります。

4. サービスが VERIFYING 状態にある場合、エージェントは、該当ホスト上にサービスのインタフェース名を持つすべてのサーバプロセスのリストをブローカに照会します。

   エージェントは、サーバプロセスごとに PID を取り出し、エージェントのサービスリスト中の PID リストと比較します。エージェントがサーバプロセスを ping することができず、エージェントの初期化ファイルにおいて AMAGENT_CLEAN 属性を 1 に設定してある場合は、エージェントは、ネーミング・サービスからそのサーバプロセスを削除します。エージェントは、該当サービスおよびエージェントのリストにある PID を持つサーバプロセスの状態を PINGED に設定します。

5. サービスが NEW 状態にある場合、つまり、サービスが追加されたばかりの場合、エージェントはサービスレコードを初期化し、サービスの起動を試みます。

   サービスが起動した場合、エージェントはサービスの状態を STARTED に設定します。OS のエラーによってサービスが起動できない場合は、エージェントはサービスの状態を START_FAILED に設定し、起動試行カウンタをインクリメントし、モニタに状態を更新するよう要求を送ります。

6. サービスが FAILED 状態にある場合、つまり、管理サイクルの前回のループ時にエージェントがサービスの ping に失敗した場合、エージェントは起動試行カウンタをチェックします。

起動試行回数が試行許可回数の最大値に等しい場合、エージェントはサービスの状態を MAXED_OUT に設定し、モニタに状態を更新するよう要求を送ります。等しくない場合は、エージェントはサービスの起動を試み、サービスの状態を STARTED に設定します。

7. サービスの状態のチェックが完了すると、エージェントはモニタからの RPC 要求をチェックします。

要求が存在する場合、エージェントは、要求を 1 つ処理した後、管理サイクルに戻り、次のサービスの状態をチェックします。

# 第 3 章 AppMinder の起動

この章では、**AppMinder** のモニタ、エージェントおよびビューアを起動する方法について説明します。また、システム上で必要となる **AppMinder** ファイルおよび **AppMinder** を実行させるために設定しなければならないオプションについても解説します。

## 起動の前に

この節では、システムに常駐させておかなければならない **AppMinder** のファイルおよび **AppMinder** を動作させるために設定しなければならないオプションについて解説します。

ファイルの必要条件はプラットフォームによって変わります。

### Windows

Windows 上で **AppMinder** を動作させるには、PATH で指定しているいずれかのディレクトリに次のファイルがなければなりません。

| ファイル名 | 説明 |
|---|---|
| ammon.exe | モニタプログラム |
| appmagt.exe | エージェントプログラム |
| amlodsvr.exe | サーバをロードするコマンド |
| amuldsvr.exe | サーバをアンロードするコマンド |
| libamclient.dll | **AppMinder** クライアントライブラリ |
| libamclnutl.dll | **AppMinder** クライアントユーティリティライブラリ |
| libamcmn.dll | **AppMinder** の共通ライブラリ |
| amviewer.exe | ビューアプログラム |
| agtlaunch.exe | ラウンチャープログラム |
| pmond.exe | プロセスモニタプログラム |
| librpc.dll | Nextra RPC ライブラリ |
| appmmsg.cat | メッセージカタログ |

## すべての UNIX プラットフォーム

**AppMinder** を動作させるには、PATH で指定しているいずれかのディレクトリに次のファイルがなければなりません。

| ファイル名 | 説明 |
|---|---|
| ammon | モニタプログラム |
| appmagt | エージェントプログラム |
| amlodsvr | サーバをロードするコマンド |
| amuldsvr | サーバをアンロードするコマンド |
| appmmsg.cat | メッセージカタログ |

# AppMinder モニタの起動

この節では、モニタの起動方法について説明します。モニタを起動する前に、モニタが登録されるモニタブローカが動作中であることを確認してください。モニタを起動してからでないと、**AppMinder** のユーティリティコマンドを発行することができません。コマンドによっては、エージェントも起動しておかないと発行できないものもあります。**AppMinder** モニタを起動するコマンドは次のとおりです。

ammon -c *initialization_file* [-i] [-p *password*] [-v] [-h] [-b *broker_key*][-m]

表 3.1 : AppMinder モニタのコマンドラインオプション

| オプション | 説明 |
|---|---|
| -c *initialization_file* **必須** | モニタ動作を制御する属性と、その対応値で構成されるテキストファイル。 |
| -i | 初期化ファイルのサンプルを ammon.ini という名前で生成します。このファイルには、すべての属性のデフォルト値が入っており、テンプレートとして使用することができます。 |
| -p *password* | パスワード。はじめてモニタを起動するときに指定しなければなりません。モニタ/エージェント間の通信には、同じパスワードを使用しなければなりません。パスワード指定がない場合、初期化ファイルの AMMON_PASSWD 属性で指定されたパスワードが使用されます。初期化ファイルにパスワードがない場合、コマンドラインでパスワードを指定しなければなりません。コマンドラインでパスワードを指定すると、モニタは、そのパスワードを暗号化して初期化ファイルに書き込みます。初期化ファイルにパスワードがある場合、モニタは、このパスワードを、コマンドライン |

| | |
|---|---|
| | で指定されたパスワードを暗号化したもので置きかえます。 |
| -v | バージョン情報を表示します。 |
| -h | コマンドラインオプションのリストを表示します。 |
| -b | 現在サポートされていません。 |
| -m | スレッド起動。<br><br>環境変数にて DCE_DROP_AGT_RPC を設定し、モニタのパフォーマンスを改善することができます。但し、スレッド起動された場合にのみ有効です。デフォルトでは 0 が設定されおり、この場合、モニタが他の RPC リクエストを処理中であった場合は、エージェントからの RPC リクエストは処理しません。 |

表 3.2 に、初期化ファイル内の属性値の説明を示します。初期化ファイルの命名法に制約はありませんが、.ini というファイル拡張子を指定すると良いでしょう。ファイル拡張子を共通にすることで、トラブルシューティングをより簡単に実行することができます。初期化ファイルには空白行を入れないでください。値の割り当てには次の形式を使用します。

たとえば、

AMMON_MODE=1

表 3.2：モニタ初期化ファイルの設定

| 属性 | 値/説明 | デフォルト値 |
|---|---|---|
| AMMON_ACTIVE | モニタが既にロードし、現在管理している設定のリストが入っています。モニタ稼働中は編集不可フィールド。 | なし |
| AMMON_AGTPORT | エージェントが動作するポートの番号。 | 7001 |
| AMMON_AGTTIMEOUT | エージェントを failed とマークし、該当ホスト上のすべてのサービスを unreachable とマークする前に、モニタがエージェントから通信してくるのを待つ時間。単位は秒。 | 300 |
| AMMON_ID | 初期化ファイルをはじめて読み出すときにモニタが生成する識別子。編集不可フィールド。 | なし |
| AMMON_LOG | モニタが状態とエラー情報を書き込むログファイルの名前。 | monitor.log |
| AMMON_LOGLEVEL | AMMON_LOG 属性で指定されたログファイルにモニタが書き | NORMAL |

| | | |
|---|---|---|
| | 込むときのログ情報のレベルを指定します。<br><br>NORMAL=エラーと重大な動作のみ<br>DEBUG=すべての動作 | |
| AMMON_LOGSIZE | AMMON_LOG 属性で指定されたログファイルの最大サイズを制御します。 | 1MB |
| AMMON_MODE | モニタのサーバモードを制御します。<br><br>0=TCP | 0 |
| AMMON_PASSWD | 暗号化されたモニタのパスワード。この値を初期化ファイルの中で編集することはできません。-p コマンドラインオプションでパスワードを指定することで、この値を設定することができます。 | なし |
| AMMON_SERVERARGS | モニタが起動時に使用する環境ファイルの名前。 | -e ammon.env |
| AMMON_TRPCLOG | モニタからエージェントへの通信についての情報を書き込むためのログファイルの名前を指定します。 Nextra では使用されません。 | monitrtcp.log |
| AMMON_TRPCLOGLEVEL | ログ情報のレベルを指定します。このフィールドは、DCE_DEBUGLEVEL 環境ファイル属性で設定できるものと同じ値をサポートします。これらの値の詳細は『リファレンス』を参照。<br><br>NONE<br>ERROR<br>WARNING<br>DEBUG | ERROR,<br>DEBUG |

# AppMinder エージェントの起動

この節では、エージェントの起動方法について説明します。

**AppMinder** エージェントプロセスを起動するコマンドは次のとおりです。

appmagt -c *initialization_file* [-pname <*pipe name*>][-i] [-p *password*] [-recover] [-v] [-h] [-repair]

表 3.3： AppMinder エージェントのコマンドラインオプション

| オプション | 説明 |
|---|---|
| -c *initialization_file*<br><br>必須 | エージェント設定時に使用する属性とその対応値で構成されるテキストファイル。 |
| -p *pipe name*<br>Windows のみ | 1Windows システム上で、複数セットの管理構成をとる場合に、かつ 2 つ目以降のエージェントを起動をする際に指定が必要になります。名称には、アルファベット、数字、そして_(アンダースコア)が使用できます。 |
| -i | 初期化ファイルのサンプルを appmagt.ini という名前で生成します。このファイルにはすべての属性のデフォルト値が入っており、テンプレートとして使用することができます。 |
| -p *password* | パスワード。はじめてエージェントを起動するときに指定しなければなりません。モニタ/エージェント間の通信には、同じパスワードを使用しなければなりません。パスワード指定がない場合、初期化ファイルの AMAGENT_PASSWORD 属性で指定されたパスワードを使用します。初期化ファイルにパスワードがない場合、コマンドラインでパスワードを指定しなければなりません。コマンドラインでパスワードを指定すると、エージェントは、そのパスワードを初期化ファイルに書き込みます。初期化ファイルにパスワードがある場合、エージェントは、このパスワードを、コマンドラインで指定されたパスワードに変更します。 |
| -recover | エージェントが前回動作を停止したときに動作状態であったかもしれないサーバをローカル設定 DB の中で探すことをエージェントに指示します。通常、エージェントが動作を停止するときは、エージェントは自分が管理しているサーバも停止させます。ただし、エージェントの動作の停止後もサーバが動作したままになることがあります。-recover の指定時に稼働中のサーバをエージェントが見つけると、エージェントはそれらのサーバを管理します。それ以外の場合、エージェントは別のサーバを起動します。UNIX 版のみで作動。 |
| -repair | .cdb ファイルを新規作成します。既に存在する場合は削除します。<br>-recover と同時に指定した場合は -recover を優先します。 |

| | |
|---|---|
| -v | バージョン情報を表示します。 |
| -h | コマンドラインオプションのリストを表示します。 |

表 3.4 は、初期化ファイルの中の値を割り当てる対象属性を示します。初期化ファイルの命名法に制約はありませんが、.ini というファイル拡張子を指定するとよいでしょう。ファイル拡張子を共通にすることで、トラブルシューティングをより簡単に実行することができます。初期化ファイルには空白行を入れないでください。値の割り当てには次の形式を使用します。

たとえば、

AMAGENT_PORT=7500

表 3.4：エージェント初期化ファイル設定

| 属性 | 値/説明 | デフォルト値 |
|---|---|---|
| AMAGENT_BROKER_SYNC_PERIOD | エージェントが管理するサーバに対して、Broker（ブローカ）への位置情報再登録を行う実行周期を指定します。単位は秒。<br><br>AMAGENT_MGMTINTERVAL より短い時間を指定しますと、AMAGENT_MGMTINTERVAL の間隔で作動しますので、ご注意ください。 | 3600 |
| AMAGENT_BURST_TIME_FOR_SEND | AMAGENT_MIN_UPD_FOR_SEND 属性で指定された値に達した後、エージェントからモニタにステータス情報を送信する間隔を指定します。単位は秒。<br><br>モニタ初期化ファイルの属性 AMMON_AGTTIMEOUT で指定された値より小さい値を必ず指定してください。 | 60 |
| AMAGENT_CLEAN | エージェントがデッドサーバエントリをネーミング・サービスから削除するかどうかを指定します。エージェントがサーバを ping することができなければ、そのサーバは停止しているとみなされます。このオプションを 1 に設定すると、アプリケーションの性能が改善されます。<br><br>0=ネーミング・サービスから削除しない。 | 1 |

| | | |
|---|---|---|
| | 1=ネーミング・サービスから削除する。 | |
| AMAGENT_DCE_CLN_TIMEOUT | エージェントからのブローカ（Broker）を含むサーバへの RPC リクエストの戻りを待つ秒数を指定します。指定秒数が過ぎても結果が戻らなければ、エージェントは接続を切断して、ハングしないようにします。最大値は *LONG_MAX* です。単位は秒。 | 5 |
| AMAGENT_DCE_SVR_TIMEOUT | エージェントに対してモニタが接続した後、エージェントが RPC を待つ秒数を指定します。指定秒数が過ぎても RPC が到着しないと、エージェントはモニタへの接続を切断してハングしないようにします。最大値は *LONG_MAX* です。単位は秒。 | 30 |
| AMAGENT_CONFIGDB | エージェント設定 DB を格納するファイルの名前。既存ファイルを指定しないと、エージェントは、空の設定 DB を作成します。 | appmagt.cdb |
| AMAGENT_DEBUG | AMAGENT_LOG 属性に割り当てるログファイルにエージェントが書き込む情報のログレベルを指定します。<br><br>0=エラーと重大な動作のみ<br>1=すべての動作 | 0 |
| AMAGENT_LOG | エージェントがログ情報を書き込むときの宛先ファイルの名前を指定します。AMAGENT_DEBUG 属性でログ情報のレベルを設定します。 | appmagt.log |
| AMAGENT_LOGSIZE | エージェントログファイルの最大サイズを指定します。 | 1MB |
| AMAGENT_MIN_UPD_FOR_SEND | ステータス情報最低更新回数が指定され、エージェントが管理サイクル内にこの値に達すると、エージェントからモニタにステータス情報が送信されます。 | 10000 |
| AMAGENT_MIN_UPD_TIME_FOR_SEND | エージェントからモニタにステータス情報を送信する間隔を指定します。単位は秒。<br><br>モニタ初期化ファイルの属性 AMMON_AGTTIMEOUT で指定された値より小さい値を必ず指定してください。 | 120 |
| AMAGENT_PASSWORD | 暗号化されたエージェントのパスワード。この値は初期化ファイルの中では編集不 | |

| | | |
|---|---|---|
| | 可です。-p コマンドラインオプションでパスワードを指定することで、この値を設定することができます。 | |
| AMAGENT_PMOND_STOP_TIME | 4 桁で指定される値は、pmond サブプロセスのリブート時間を指定します。<br>例　1315 | 0115 |
| AMAGENT_PING_PERIOD | エージェントが管理する Broker（ブローカ）／サーバに対して、PING を行う実行周期を指定します。単位は秒。<br><br>AMAGENT_MGMTINTERVAL より短い時間を指定しますと、AMAGENT_MGMTINTERVAL の間隔で作動しますので、ご注意ください。 | 120 |
| AMAGENT_MGMTINTERVAL | エージェントによる管理動作の実行周期を指定します。単位は秒。 | 120 |
| AMAGENT_PORT | エージェントが動作するポートの番号。モニタの初期化ファイル内で AMMON_AGTPORT 属性が設定されているのと同じ番号に設定しなければなりません。 | 7001 |
| AMAGENT_SHUTDOWN<br>＊未使用 | エージェントがシャットダウンするときに、エージェントが管理するすべてのサーバをエージェントがシャットダウンするかどうかを指定します。0 に設定した場合は、次にエージェント起動するときに、かならず<br>-recover を指定し、オーファンサーバが発生するのを回避してください。<br><br>0=サーバをシャットダウンしない<br>1=サーバをシャットダウンする | 1 |
| AMAGENT_TRPCLOG | RPC の通信ログファイルの名前を指定します。 | agenttcp.log |
| AMAGENT_TRPCLOGLEVEL | RPC のログ情報のレベルを指定します。このフィールドは、DCE_DEBUGLEVEL 環境ファイル属性で設定できるものと同じ値をサポートします。これらの値の詳細は『リファレンス』を参照してください。<br><br>NONE<br>ERROR<br>WARNING<br>DEBUG | NONE,<br>NONE |

# ビューアの起動

ビューア、つまり **AppMinder** のグラフィカル・ユーザ・インタフェースを起動する前に、「起動の前に」に示したファイルを必ずシステムに常駐させ、必要な環境変数を設定してください。また、設定のロードやサービスの起動・停止を行う場合、接続先とするモニタおよびエージェントを必ず起動してください。そのあと、次のコマンドを入力します。

> amviewer [-h *monitor_host*] [-p *monitor_port*]

ビューアのメインウィンドウと、トランスポートセキュリティログインのダイアログ・ボックスが画面に現れます。

ただし、-h -p オプションを起動時に指定し、かつコンフィギュレーションがアクティブであるモニタに接続した場合、ビューアは選択画面を表示せず、モニタのアクティブ設定の全リストがビューアのメインウィンドウに表示されます。

# 第 4 章 AppMinder ビューアの使用

この章では、ビューアとして知られている **AppMinder** の GUI を使って設定を管理する方法について説明します。

## モニタとの接続

ビューアを起動すると、図 4.1 に示すようなビューアのメインウィンドウと Connect To Monitor ダイアログ・ボックスが現れます。

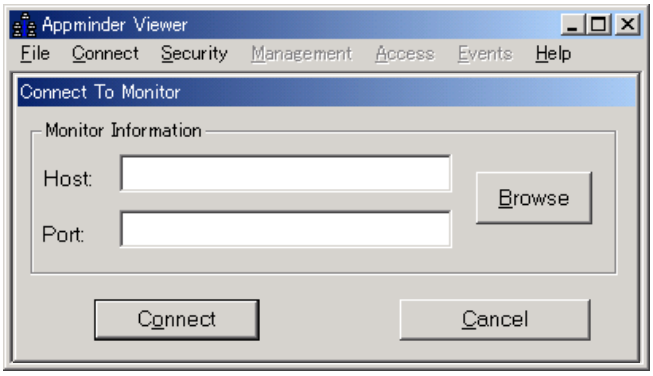


図 4.1： Connect To Monitor ダイアログボックス

モニタのホストとポートがわからない場合は、Browse ボタンをクリックして環境ファイルを検索してください。図 4.2 のような File Selection ダイアログ・ボックスが現れます。

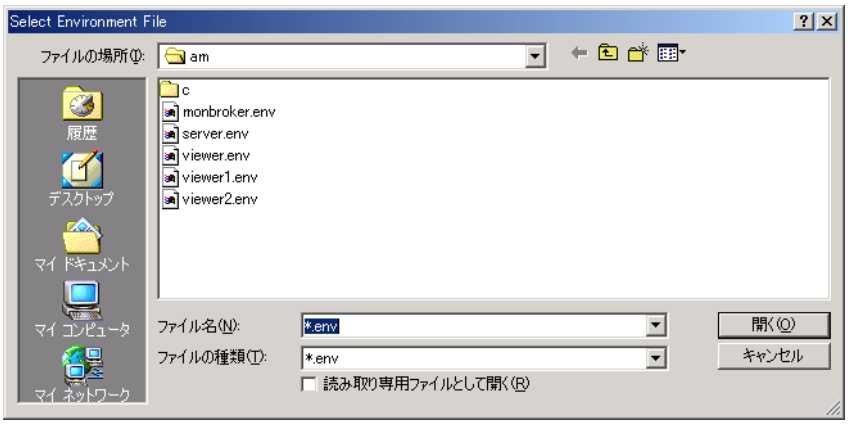


図 4.2： File Selection ウィンドウ

File Selection ウィンドウでは、ファイルシステムの中の環境ファイルをフィルタによって検索することができます。Viewer 用の環境ファイルを選択し、OK ボタンをクリックすると、図

4.3 のような Monitor Selection ウィンドウが画面に現れ、その中に指定したモニタブローカに登録されているモニタの一覧が表示されます。

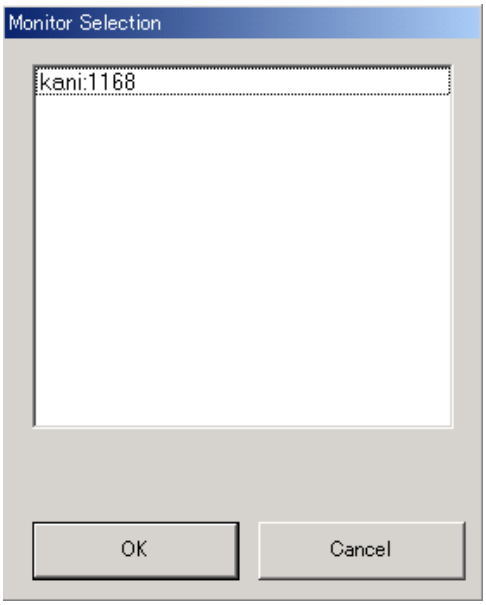


図 4.3： Monitor Selection ウィンドウ

Viewer の接続先とするモニタのホストとポート番号を選択し、OK ボタンをクリックしてください。選択したホストとポート番号が Monitor Connect ダイアログ・ボックスに現れます。Connect ボタンをクリックし、選択したモニタに接続してください。

# 設定の検索

モニタに接続すると、ビューアのメインウィンドウには、図 4.4 のような、該当モニタのアクティブ設定全部を示すリストが自動的に表示されます。

アクティブ設定がない場合は、リストは表示されません。

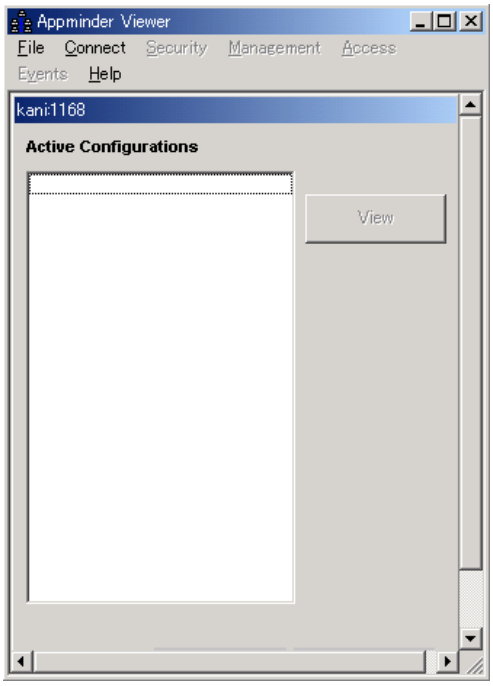


図 4.4：アクティブ設定を示すビューアのメインウィンドウ

表 4.1 から表 4.6 にビューアのメインウィンドウ・プルダウンメニューから選択できるオプションについてまとめます。

## File メニュー

表 4.1：File メニューのオプション

| オプション | 説明 |
|---|---|
| New | 空の設定ファイルを新規に作成します。 |
| Open... | Remote File 選択ウィンドウを表示します。ファイルシステムの中の設定ファイルを検索することができます。 |
| Save | 設定ファイルをディスクに書き込み、アクティブ設定またはスタンバイ設定に行った変更内容を保存します。注意: アクティブ設定からサービスを追加または削除すると、ただちに変更されます。ただし、AppMinder を閉じる前に設定ファイルを保存しない限り、その変更内容はディスクに書き込まれません。アクティブ設定ファイルを編集する場合、変更内容は、ただちには有効になりません。有効にするには、変更内容を保存し、設定をアンロードした後、設定をロードしなければなりません。 |
| Save As... | 設定ファイルを別の名前で保存します。 |

| | |
|---|---|
| Close | 設定ファイルを閉じます。注意: アクティブ設定ファイルを閉じても、設定の状態は変更されません。 |
| Exit | ビューアを終了します。 |

## Connect メニュー

表 4.2：Connect メニューのオプション

| オプション | 説明 |
|---|---|
| Open Connection To... | 接続先とするモニタのホストとポート番号を指定するダイアログ・ボックスを表示します。 |
| Disconnect | モニタとのビューアセッションを終了します。注意: モニタから切断してもアクティブ設定には影響しません。設定を管理するモニタやエージェントが動作中である限り、設定はアクティブの状態を保ちます。 |

## Security メニュー

表 4.3： Security メニューのオプション

| オプション | 説明 |
|---|---|
| Set Monitor Password | セッションセキュリティ用に、パスワードを指定するダイアログ・ボックスを表示します。 |

## Management メニュー

表 4.4：Management メニューのオプション

| オプション | 説明 |
|---|---|
| Begin Managing | サービスリストに示されている設定をロードして、アクティブにします。 |
| Stop Managing | アクティブな設定を停止します。 |

## Event メニュー

表 4.5： イベント処理メニューのオプション

| オプション | 説明 |
|---|---|
| Binding List | 該当設定のモニタで現在登録されているイベントハンドラーサーバのリストを表示します。 |

## Help メニュー

表 4.6：Help メニューのオプション

| オプション | 説明 |
|---|---|
| About AM Viewer | ビューアのバージョンを示します。 |

アクティブ、スタンバイに関係なく設定の全リストを参照するには、File プルダウンメニューから Open を選択してください。図 4.5 に示すようなファイル選択ウィンドウが現れます。ここから特定ディレクトリの中の設定ファイルを検索することができます。また、ファイルシステムがドライブ名の指定に対応している場合、ディスクドライブ名で設定ファイルを検索することができます。

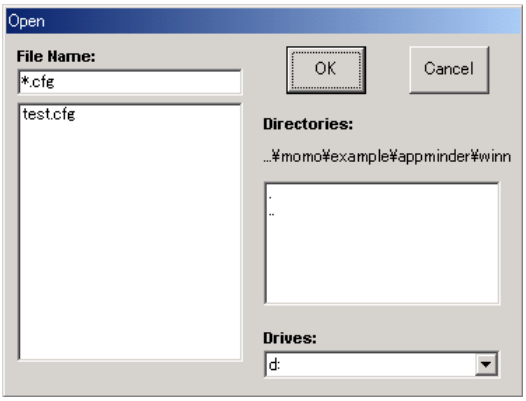


図 4.5： 設定ファイル選択ウィンドウ

参照しようとする設定を探すときは、設定を選択し、OK ボタンをクリックしてください。ファイル選択ウィンドウが閉じ、設定ファイルの名前がビューアのメインウィンドウに現れます。

設定ファイルの階層ツリービューを参照するには、ビューアのメインウィンドウで設定ファイルをダブルクリックしてください。図 4.6 に示すような設定では、階層ツリーにブローカ、サブブローカおよびサーバプロセスが表示されます。

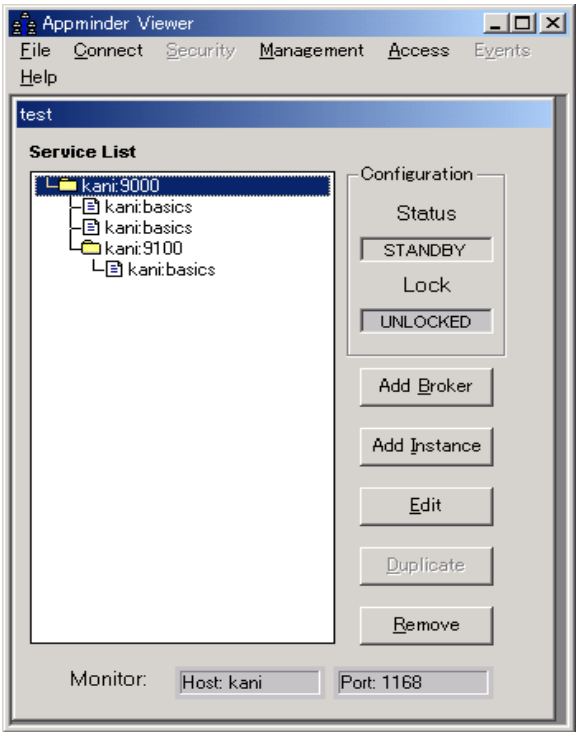


図 4.6：設定に対応するサービスの階層

# 設定の更新

この節では、ビューアから設定を更新する方法について説明します。特に以下の事項について説明します。

- ビューアのメインウィンドウでのボタンの使い方
- アクティブ状態ウィンドウでのボタンの使い方
- 設定の編集

## ビューアのメインウィンドウの操作

図 4.6 に示すように、ビューアのメインウィンドウでサービスリスト名をダブルクリックすると、ビューアは、その設定のすべてのサービスを階層ツリーの形式で表示します。特定のサービスに操作を行うには、階層ツリーで、そのサービスを選択し、該当する項目ボタンをクリックしてください。

アクティブ設定からブローカまたはサーバプロセスを追加または削除した場合は、モニタは、その変更内容をディスクに保存しません。新規の設定ファイルをディスクに書き込むには、File プルダウンメニューから Save または Save As を選択してから設定を閉じてください。

## ブローカの追加

ブローカを追加するには、Add Broker ボタンをクリックしてください。ダイアログ・ボックスがポップアップします。

サービスをアクティブ設定に追加すると、ビューアはアクティブ状態ウィンドウにサービスを表示し、エージェントはサービスの起動を試みます。

## プロセスの追加

サーバプロセスを設定に追加するには、階層ツリーの中のブローカを選択して Add Instance ボタンをクリックしてください。ダイアログ・ボックスがポップアップし、そのブローカに登録する新規のサーバプロセスを指定することができます。設定がアクティブな場合、ビューアは、新規のサーバプロセスのリストをアクティブ状態ウィンドウに表示し、エージェントは該当するサーバプロセスの起動を試みます。

## サービスの削除

サービスを削除するには、階層ツリーでそのサービスを選択し Remove ボタンをクリックしてください。ブローカの削除を実行すると、エージェントは、そのブローカの他に、そのブローカに登録されているサーバプロセスとサブブローカを停止、削除します。設定がアクティブな場合、サービスは、アクティブ状態ウィンドウ上で UNLOADED に変わります。

## ロードとアンロード

Management メニューから Begin Managing を選択し、設定の中のサービスをすべて起動します。設定をロードすると、状態がスタンバイからアクティブに変わります。最初に設定をロードしておかないと、その設定の中の特定のサービスを停止および再起動することができません。

一旦設定をロードしたら、設定を閉じ、ビューアをモニタから切断し終了することができます。そのとき、モニタ、エージェントおよびサービスは起動したままです。

サービスを停止するには Stop Managing を選択します。設定をアンロードすると状態がアクティブからスタンバイに変わります。

## アクティブ状態ウィンドウの操作

はじめてモニタに接続すると、そのモニタのアクティブ設定の全リストがビューアのメインウィンドウに表示されます。View ボタンをクリックすると、ビューアは、ビューアのメインウィンドウの隣にアクティブ状態ウィンドウを開きます。アクティブ状態ウィンドウに、設定中のすべてのサービスの状態が表示されます。図 4.7 にアクティブ状態ウィンドウを示します。

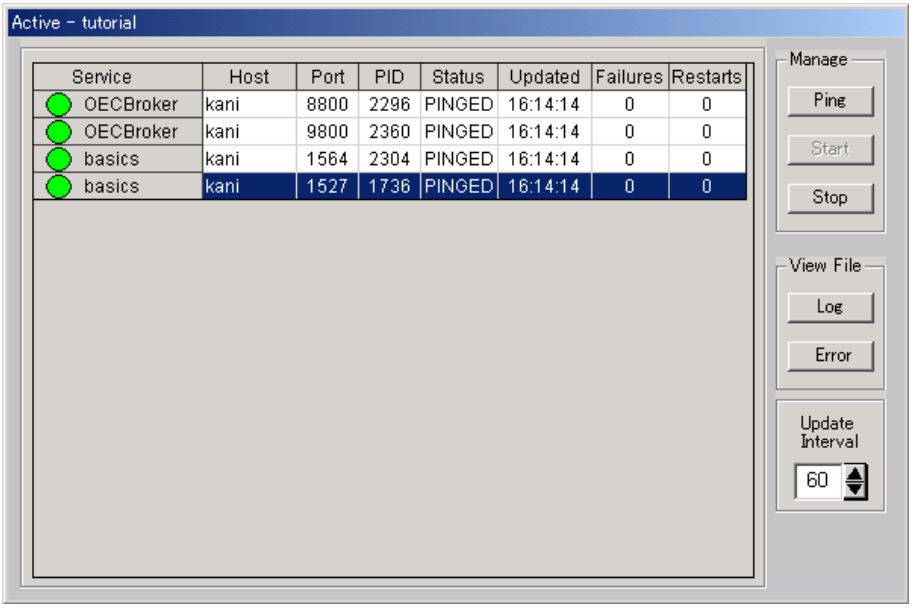


図 4.7：アクティブ状態ウィンドウ

モニタが最新の状態情報をビューアに送り、ビューアが新規情報をアクティブ状態ウィンドウに表示します。デフォルトでは、モニタは 60 秒ごとに更新された状態情報をビューアに送ります。この設定値は Update Interval ボタンで変更することもできますが、ネットワークトラフィックが増し、パフォーマンスが低下する原因となりますので、この値を下げないほうが良いでしょう。

サービスに対してピング、起動、停止を行うには、グリッド内でサービスを選択した後、ウィンドウの右側にある該当するボタンをクリックしてください。

ブローカを起動または停止すると、エージェントは以下のサービスの起動、停止を行います。

- ブローカ
- ブローカに登録されているすべてのサーバプロセス
- ブローカのすべてのサブブローカ
- サブブローカに登録されているすべてのサーバプロセス

## 設定の編集

設定の中のサービスを編集するには、階層ツリーで選択し、Edit ボタンをクリックしてください。ビューアのメインウィンドウの隣に、図 4.8 に示すような Edit ウィンドウが現れます。このウィンドウを使って、サービスを起動するのに必要な情報を指定してください。図 4.8 に、例としてブローカを起動するコマンドの場合を示します。

ウィンドウの最上部のタブをクリックすると、それぞれの設定を編集できます。表 4.7 に個々の Edit ページについてまとめます。

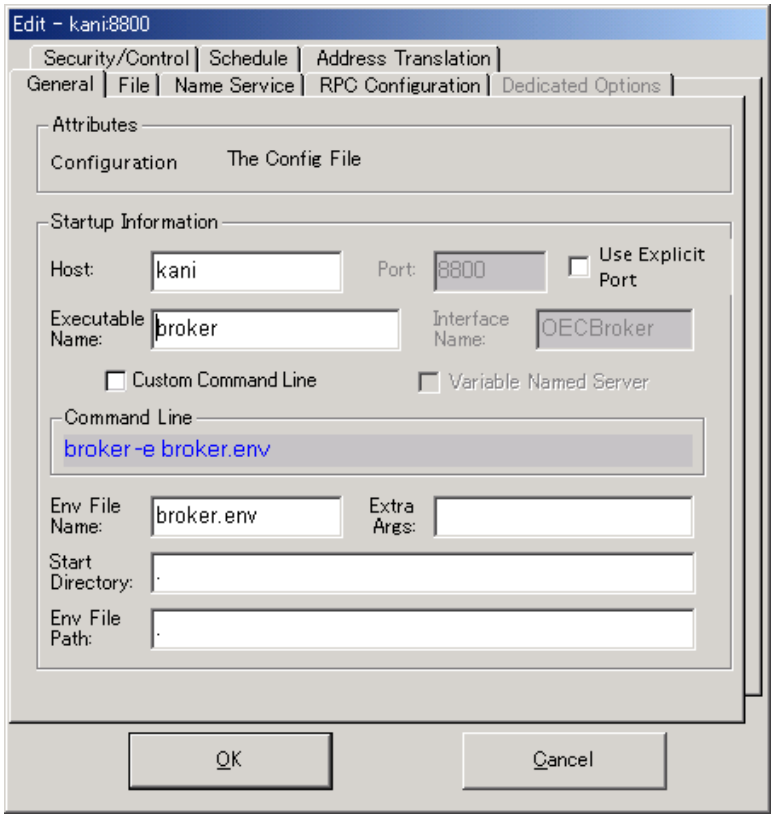


図 4.8：Edit ウィンドウ

表 4.7：Edit タブ説明

| タブ説明 | 目的 |
|---|---|
| General | サービスを起動するのに必要な情報を指定します。 |
| File | サービスの環境ファイルの名前と位置、エラーファイルおよびログファイルとログに書き込む情報のレベルを指定します。 |
| Name Service | 使用するネーミング・サービスを指定します。ブローカリストと拡張ブローカリストを指定することができます。 |
| RPC | パケットサイズやタイムアウト値など、コンポーネント間の RPC 通信を制御するための属性を設定します。 |
| Dedicated Server | クライアント数の最大値など、デディケイテッド・サーバオプション |

| | |
|---|---|
| | を設定します。 |
| Security/Control | トランスポートセキュリティおよび ping 遅延時間や最大試行回数などの制御情報を設定します。 |
| Schedule | サービスの起動停止時刻をスケジューリングすることができます。 |
| Address Translation | アドレス変換の設定ができます。詳しくは『リファレンス』にある環境ファイル属性 DCE_TRANSCRIPT の説明を参照してください。 |

Edit で設定することのできる属性のほとんどは、環境ファイルの属性に対応しています。これらの属性を設定することによる影響については、『リファレンス』を参照してください。

アクティブ設定を編集した場合、変更内容はただちに有効にはなりません。設定をアンロードし、ロードし直すことで有効になります。

# プロセスのステータスシンボル

表 4.8 は、AmViewer でプロセスのステータスを表すシンボルについて説明します。

表 4.8：ステータスシンボルの説明

| シンボル | イベント | 16 進数値 | 説明 |
|---|---|---|---|
|  三角 – 黄 | AM_NEW | 0x000001 | 新しいエントリが DB に追加されましたが、まだ起動されていません。 |
| | AM_STARTING | 0x000002 | プロセス起動を開始しています。 |
| | AM_STARTED | 0x000003 | プロセスが正常に起動されました。 |
| | AM_RESTART | 0x000010 | 再起動するように設定されました。 |
|  三角–黄＆緑 | AM_VERIFYING | 0x000400 | 確認の為に、プロセスより情報を入手しています。 |
|  球 – 黄 | AM_BUSY | 0x001000 | RPC の応答に失敗しました。 |
| | AM_DEDMAXED | 0x002000 | サーバは最大数のクライアントを処理しています。 |

| | | | |
|---|---|---|---|
| | AM_REREGISTER | 0x040000 | ブローカで何らかの失敗があったため、再登録が必要です。 |
|  三角 – 緑 | AM_PINGING | 0x000100 | プロセスは ping されています。 |
|  球 – 緑 | AM_PINGED | 0x000200 | ping に成功しました。プロセスは RPC を処理します。 |
| | AM_VERIFIED | 0x000800 | エージェントによって起動されたプロセスだということが確認されました。 |
|  球 – 青 | AM_STOP | 0x080000 | プロセスは停止されます。休止ステートに移ります。 |
| | AM_IDLE | 0x100000 | プロセスは休止状態です。 |
|  球 – 赤 | AM_STARTFAIL | 0x000008 | 起動に失敗しました。 |
| | AM_MAXED | 0x000020 | 再起動が最大数回行われました。 |
| | AM_LOST | 0x008000 | ホストへの接続がなくなりました。 |
| | AM_FAILED | 0x003000 | ping に失敗しました。 |
| | AM_OVERLOADED | 0x010000 | プロセスの負荷がとても高い状態です。 |
| | AM_UNREACHABLE | 0x400000 | このプロセスを管理しているエージェントはモニタと通信していません。 |
|  球 – 黒 | AM_UNMANAGED | 0x000040 | プロセスは AppMinder の管理外です。 |
| | AM_ADOPTED | 0x000080 | 現在は AppMinder に管理されています。 |
| | AM_BACKUP | 0x020000 | これは予備プロセスです。 |
| | AM_REMOVE | 0x200000 | プロセスは設定ファイル(.cfg ファイル)からアンロードまたは削除されました。 |

# 第 5 章 AppMinder のコマンドラインインタフェースの使用

この章では、**AppMinder** コマンドラインインタフェース(`amlodsvr` と `amuldsvr`)を使って、アプリケーション設定の管理を行う方法について説明します。

## 概要

この節では、**AppMinder** コマンドラインインタフェース(`amlodsvr` と `amuldsvr`)に共通な引き数について説明します。

### 共通引き数と環境変数

この節では、すべての **AppMinder** コマンドに共通な引き数について説明し、コマンドラインに指定する引き数の数を減らすことのできる環境変数について解説します。

表 5.1 に共通引き数についてまとめてあります。次節以降では各コマンドに特有の引き数だけを説明します。

表 5.1：コマンドライン共通引き数

| 引き数 | 説明 |
|---|---|
| -h *monitor_host* | モニタが動作するホストマシンの名前を指定します。 v5.2 以降では、IP アドレスでの指定も可能です。 |
| -p *user_password* | ユーザのパスワードを指定します。この引き数は、セキュリティのかかった RPC 通信を実行する場合にだけ指定します。 |
| -m *monitor_password* | モニタのパスワードを指定します。セキュリティモード(モニタ初期化ファイルに AMMON_SECURE=1 を指定する)でモニタを起動する場合、この引き数を指定しなければなりません。 |
| -e *environment_file* | 環境ファイルのファイル名を指定します。 |
| -c *drive_path_and_file* | ディスクドライブ名(および必要な場合、設定ファイルのパス |

| | |
|---|---|
| | とファイル名)を指定します。設定ファイルの命名法には制約はありませんが、本書では.cfg というファイル拡張子を指定しています。ファイル拡張子を共通にすることで、より簡単にトラブルシューティングを実行することができます。 |
| -? | コマンドについてのヘルプメッセージを表示します。 |
| -v | バージョン情報を表示します。 |

コマンドラインに指定する引き数の数を減らすために、表 5.1 に示した引き数に以下の環境変数を設定することができます。

- APPMMONHOST=*monitor_host*
- APPMUSRPWD=*user_password*
- APPMENVFILE=*environment_file*
- APPMCFGFILE=*configration*

# サービスの起動と停止

この節では、コマンドを使って以下の操作を行う方法について説明します。

- 設定をロードすることですべてのサービスを起動する。
- 設定をアンロードすることですべてのサービスを停止する。
- 単一のサービスを起動する。
- 単一のサービスを停止する。

引き数の多くがすべてのコマンドに共通なので、次節では各コマンドに特有の引き数だけを説明します。すべてのコマンドで共通に使用する引き数については、「共通引き数と環境変数」を参照してください。

## 設定のロード

ある設定の中のすべてのサービスを起動したり、特定のサービスを起動するには amlodsvr コマンドを次のように発行します。

```
amlodsvr -h monitor_host -p user_password -e environment_file -c
drive_path_and_file [-i all| instance=uuid | service=uuid]
```

| パラメータ | 説明 |
|---|---|
| -i | 設定の中のすべてのサービス、あるいは特定の単一のサービ |

| | |
|---|---|
| | スを起動するように amlodsvr コマンドに指示します。すべてのサービスを起動するときは all を指定します。<br>特定のサービスを起動するには以下のように指定します。<br>-i instance = *uuid*<br><br>特定のブローカとその配下にあるサービス全てを起動する場合には以下のように指定します。<br>-i service=*uuid*<br><br>＊*uuid* は、コンフィグレーションファイル(*.cfg)を参照して、該当するサービスの *uuid* を指定してください。 |

このコマンドを発行するときは、あらかじめ設定のサービスを起動するためのエージェントを 1 つ以上起動しておかなければなりません。

設定ファイルに all を指定すると、amlodsvr コマンドは、すべてのブローカとサーバプロセスを起動します。

最初に設定全体をロードしておかないと、特定のサービスを起動することはできません。設定をロードした後、amuldsvr コマンドを発行することで特定のプロセスを停止することができます。そのあと、-i instance=uuid、または -i service=uuid オプションを指定した amlodsvr コマンドを発行することで、これらの特定のプロセスを再起動することができます。

設定をロードすると、設定状態がスタンバイからアクティブに変わります。設定がアクティブになったら、特定のサービスの停止または起動、およびサーバプロセスとエージェントへの ping を行うことができます。

| 戻り値 | 意味 |
|---|---|
| 0 | 正常に実行されました。 |
| それ以外 | RPC エラー番号が返されます。『トラブルシューティングガイド』「Nextra のエラーメッセージ」を参考にしてください。 |

## 設定のアンロード

ある設定の中のすべてのサービスを 7 停止したり、特定のサービスを停止するには amuldsvr コマンドを次のように発行します。

amuldsvr -h monitor_host -p user_password -e environment_file -c drive_path_and_file [-i all | instance=uuid | service=uuid]

| パラメータ | 説明 |
|---|---|

| | |
|---|---|
| -i | 設定中のすべてのサービス、あるいは特定のサービスを停止するように amuldsvr コマンドに指示します。すべてのブローカとサーバプロセスを停止するときは all を指定します。<br>特定のサービスを停止するには以下のように指定します。<br>-i instance = uuid<br><br>特定のブローカとその配下にあるサービス全てを停止する場合には以下のように指定します。<br>-i service=uuid<br><br>＊uuid は、コンフィグレーションファイル(*.cfg)を参照して、該当するサービスの uuid を指定してください。 |

設定ファイルに all を指定すると、amuldsvr コマンドは、すべてのブローカとサーバプロセスを停止します。 特定のサービスを停止する場合は、-i instance=uuid、または -i service=uuid と指定します。

設定をアンロードすると、設定状態がアクティブからスタンバイに変わります。

| 戻り値 | 意味 |
|---|---|
| 0 | 正常に実行されました。 |
| それ以外 | RPC エラー番号が返されます。『トラブルシューティングガイド』「Nextra のエラーメッセージ」を参考にしてください。 |

# AppMinder プロセスの停止

**AppMinder** プロセスを停止するときは、kill -9 コマンドを指定しないでください。
**AppMinder** は、kill -9 シグナルをトラップ(捕捉)することができず、設定ファイルの内容が壊れることがあります。**AppMinder** プロセスを停止するときは必ずソフトキルコマンドを使用してください。

# 第 6 章 チュートリアル

このチュートリアルでは **AppMinder** の使い方を示し、ブローカ、サブブローカ、2 つのサーバプロセス(`basics`)が含まれる簡単な設定を作成します。`basics` サーバの開発方法は、『サーバ開発者ガイド』を参照してください。ここでは次の項目について説明します。

## セットアップ

『**AppMinder** ユーザガイド』の「起動の前に」と **Nextra3.6** インストレーションノートを参照して、**AppMinder** がシステムに正しくインストールされているか、プラットフォームに必要な環境変数が設定されているかどうかを確認してください。このチュートリアルでは、Windows 上で AppMinder の使用方法を説明します。

UNIX 上での AppMinder の使用方法は、AmViewer を Windows で起動すること以外は同じです。

## チュートリアルのコンポーネント

図 6.1 にチュートリアルで使用するコンポーネントとそれらのキーファイルを示します。ビューア、モニタ、エージェントは、**AppMinder** のコンポーネントです。モニタはビューアにとってはサーバ、エージェントにとってはクライアントになります。ビューアがモニタの位置を確認するためには、モニタがモニタブローカに登録されていなければなりません。

エージェント以外のコンポーネントには環境ファイルを作成し、そのコンポーネントの実行時の設定を指定します。モニタとエージェントには、**AppMinder** が起動時に使う初期化ファイルも必要です。

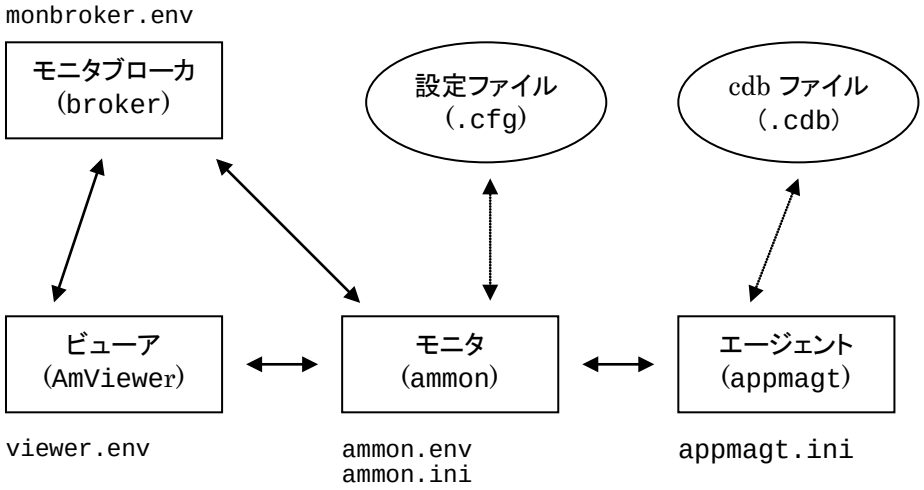


図 6.1：チュートリアルのコンポーネントとキーファイル

# 環境ファイルと初期化ファイルの作成

ここでは、ビューア、モニタ、そしてエージェントに必要な環境ファイルと初期化ファイルの作成方法を示します。

## 環境ファイルの記述

Nextra のクライアント、サーバ、ブローカには環境ファイルが必要ですが、同じようにビューア、モニタ、モニタブローカにも環境ファイルが必要です。環境ファイル中にはさまざまな属性を指定できますが、チュートリアルでは必要最小限の属性のみを指定しておきます。環境ファイルの属性については『リファレンス』を参照してください。

表 6.1 はビューア、モニタブローカ、モニタの設定に使う環境ファイルの属性を示しています。

DCE_BROKER 属性は必ず指定しなければなりません。モニタブローカの環境ファイル(monbroker.env)の DCE_BROKER 属性には、ビューアとモニタからの接続を待つブローカのホスト名とポート番号を指定します。モニタの環境ファイル(ammon.env)の DCE_BROKER 属性には、モニタが登録するブローカ(モニタブローカ)のホスト名とポート番号を指定します。ビューアの環境ファイル(viewer.env)の DCE_BROKER 属性には、ビューアがコンタクトしてモニタの位置情報を聞くブローカ(モニタブローカ)のホスト名とポート番号を指定します。

表 6.1：環境ファイルの属性の設定

| 属性 | ビューア | モニタブローカ | モニタ |
|---|---|---|---|
| DCE_BROKER= | *host*,7800 | *host*,7800 | *host*,7800 |
| DCE_LOG= | viewer.log | monbroker.log | monitor.log |
| DCE_DEBUGLEVEL= | NONE,DEBUG | NONE,DEBUG | NONE,DEBUG |

monbroker.env と ammon.env, viewer.env を編集して、DCE_BROKER 属性のホスト名を実際に使用するホスト名に変更してください。

DCE_LOG 変数にはそれぞれビューア、モニタブローカ、ブローカの名前の付いたファイル名を指定して、実行時の状態やエラー情報を記録させます。

DCE_DEBUGLEVEL 属性には、ビューア、モニタブローカ、ブローカの実行時とエラー時の情報をログファイルに記録するときの情報量を設定します。DEBUG レベルを指定すると、ランタイムオブジェクトはログファイルにエラー、警告、コンテキストメッセージを書き込みます。

## モニタ初期化ファイルの作成

モニタを起動する前に、モニタの初期化ファイルを作成しなければなりません。モニタの初期化ファイルは、モニタの動きをコントロールする属性と値の対を含むテキストファイルです。初期化ファイルを 1 から自分で記述することもできますが、次のコマンドを発行すると、ほとんどの必要な属性のデフォルトを含む ammon.ini という初期化ファイルが自動生成されます。

```
$ammon -i
```

このファイルを編集して、デフォルトの設定を変えたり、新たなエントリを追加したりします。モニタの初期化ファイルの属性については、「AppMinder モニタの起動」を参照してください。このチュートリアルでは、AMMON_SERVERARGS 属性のみを変更します。この属性にはモニタを起動するときの環境ファイルを指定します。モニタの環境ファイルが ammon.env を指すように編集してください。

```
AMMON_SERVERARGS=-e ammon.env
```


**複数セット構成の場合のモニタ初期化ファイル**

1Windows システム上で、複数セットの管理構成をとる場合には、それぞれに対して必ず 1 つの初期化ファイルが必要になります。必ず-i オプションにて初期化ファイルをそれぞれで作成、必要に応じて編集の後ご利用ください。コピーして複製を利用することは絶対に行わないでください。

### エージェント初期化ファイルの作成

モニタの初期化ファイルと同様、エージェントにも起動時に初期化ファイルが必要です。次のコマンドを発行して、ほとんどの必要な属性のデフォルトを含む `appmagt.ini` という初期化ファイルを自動生成します。

```
> appmagt -i
```

生成された初期化ファイルは編集する必要はありません。エージェントの初期化ファイルの属性については、「AppMinder エージェントの起動」を参照してください。

## AppMinder コンポーネントの起動

必要な環境ファイルと初期化ファイルが揃ったら、**AppMinder** コンポーネントを起動します。

### モニタブローカの起動

モニタを起動する前に、モニタが登録されるブローカを起動しなければなりません。次のコマンドを発行してモニタブローカを起動します。

```
> broker -e monbroker.env &
```

Windows をご使用の場合:

```
>start /b broker -e monbroker.env
```

`-e` オプションには使用する環境ファイルを指定します。`&` オプションは、モニタブローカをバックグラウンドで起動したいときに指定します。

### エージェントの起動

エージェントを起動するときには次のコマンドを発行します。

```
> appmagt -c appmagt.ini -p パスワード &
```

Windows をご使用の場合:

```
> start  /b appmagt -c appmagt.ini -p パスワード
```

-c オプションにはエージェントの初期化ファイルを指定します。エージェントを最初に起動するときは、-p オプションでパスワードを指定します。すると AppMinder がそのパスワードを暗号化してエージェントの初期化ファイルの AMAGENT_PASSWORD 属性に書き込みます。2 回目以降は、エージェントのパスワードを変更したい場合のみ、コマンドラインでパスワードを指定します。

## モニタの起動

モニタを起動するときには次のコマンドを発行します。

```
> ammon -c ammon.ini -p パスワード &
```

Windows をご使用の場合:

```
>start /b ammon -c ammon.ini -p パスワード
```

コマンドラインのシンタックスはエージェントの起動時とほぼ同じです。-c オプションには初期化ファイルを、-p オプションにはエージェント起動時と同じパスワードを指定します。エージェント同様、パスワードは初回のモニタ起動時のみ必要で、エージェントのパスワードと同じでなければなりません。モニタとエージェントは互いに通信をするときに、暗号化されたパスワードを使用します。

## ビューアの起動

Windows 上のプログラムマネジャーで、ビューアのアイコンをダブルクリックするか、ファイルマネジャー／エクスプローラで amviewer.exe ファイルをダブルクリックします。
または、コマンドラインから次のコマンドを発行してください。

```
> amviewer
```

## モニタへの接続

ビューアを使って作業をするには、まずモニタとの接続を確立しなければなりません。ここでは、すでに起動しているモニタに接続する手順を示します。

ビューアを起動すると、「Connect To Monitor」ダイアログ・ボックスが現れます。

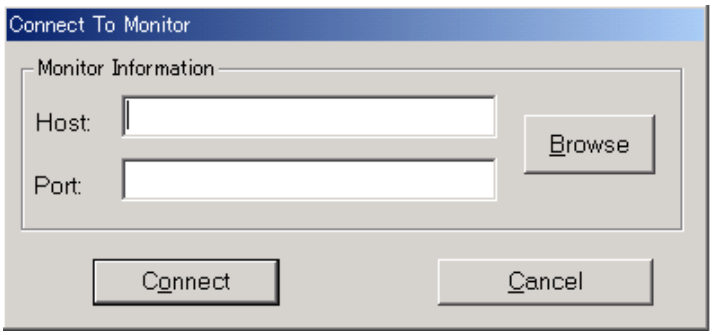


図 6.2 :「Connect To Monitor」ダイアログ・ボックス

ここで接続したいモニタのホスト名とポート番号をタイプします。ただし、ポート番号はオペレーティングシステムによって割り当てられますので、開発者がモニタのポート番号を知らない場合があります。その場合、Browse ボタンをクリックすると File Selection ウィンドウが現れます。

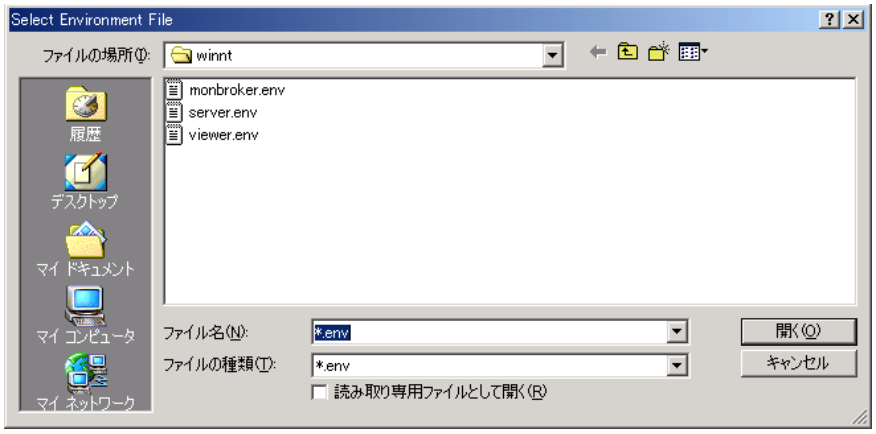


図 6.3 :「Select Environment File」ダイアログボックス

ビューアの環境ファイルをコピーしたディレクトリを検索して、ファイル(`viewer.env`)を選択しOKをクリックします。ビューアは環境ファイルからモニタが登録されているブローカを探して「Monitor Selection」ダイアログ・ボックスを表示します。

「Monitor Selection」ダイアログ・ボックスは、現在ブローカに登録されているモニタのホスト名とポート番号をリストします。このチュートリアルでは、モニタブローカに登録しているモニタが 1 つですので、1 つのホスト名／ポート番号の組み合わせしか表示されません。表示されているポート番号がモニタの環境ファイル中の `DCE_BROKER` 属性で指定したポート番号とは異なる点に注意してください。モニタの環境ファイルで指定しているのは、モニタブローカのポート番号です。「Monitor Selection」ダイアログ・ボックスに表示されるポート番号は、ビューアとエージェントからリクエストを受け付けるモニタのポート番号です。

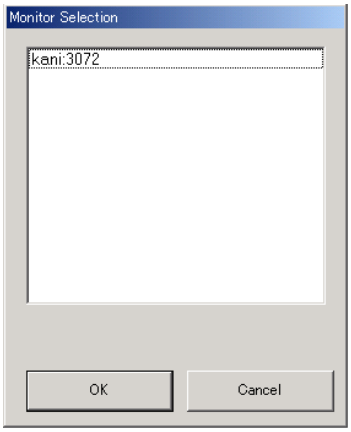


図 6.4 :「Monitor Selection」ダイアログ・ボックス

モニタのホスト名／ポート番号の対を選択してOKをクリックします。「Monitor Selection」ダイアログ・ボックスが消え、ビューアの「Connect To Monitor」ダイアログ・ボックスにモニタのホスト名とポート番号が設定されます。「Connect」ボタンをクリックしてモニタに接続します。

# 設定の作成

モニタに接続したら、そのモニタのアクティブな設定のリストがビューアのメイン画面に表示されます。

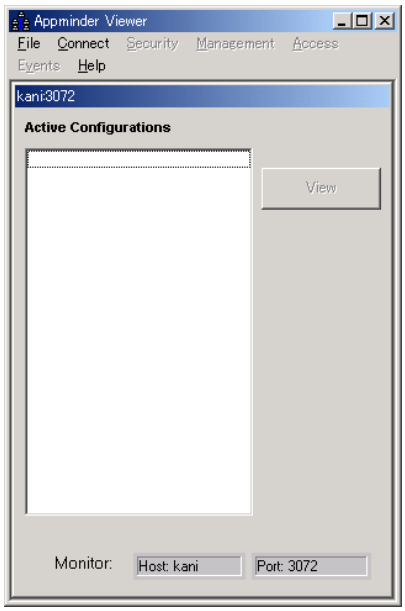


図 6.5 : アクティブ設定のリスト

まだアクティブな設定がないので、リストは空になっています。新規の設定を作成するには、File プルダウンメニューから New を選択すると、次のような「New」ダイアログ・ボックスが現れます。

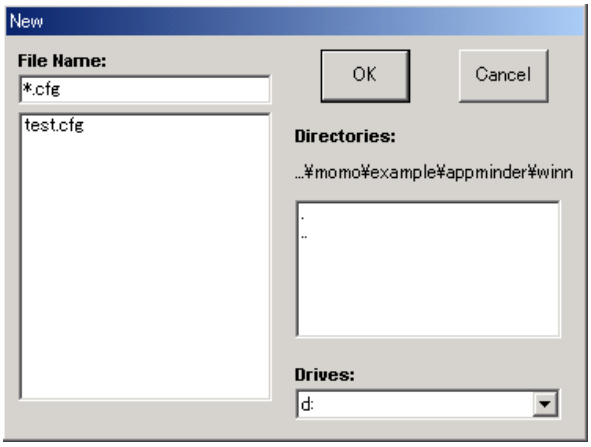


図 6.6 :「New」ダイアログ・ボックス

ファイル名のフィールドには作成したい設定の名前(tutorial)をタイプします。デフォルトのディレクトリは UNIX 上のモニタを起動したディレクトリです。ディレクトリボックスの. (ピリオド)や. . (ダブルピリオド)をクリックして、別のディレクトリに移動することもできます。OKをクリックします。これで tutorial という名前の空の設定が出来ました。次のステップは、この設定にサービスを追加していくことです。

## ブローカの追加

サーバプロセスを追加する前に、まずブローカを設定しなければなりません。

「Add Service」ボタンをクリックします。Add Service ウィンドウの「General」詳細ページが Service List ウィンドウの右側に表示されます。

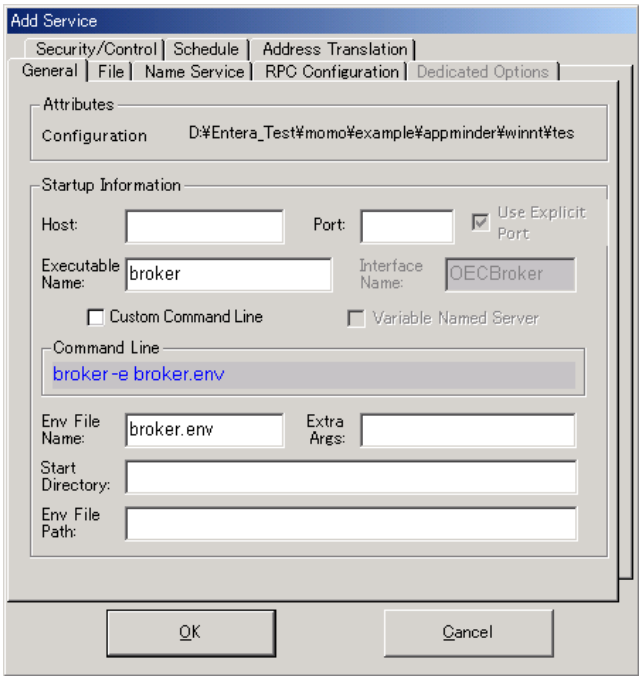


図 6.7 :「Add Service」ウィンドウ

Add Service ウィンドウにはブローカの実行時の属性を指定する 6 つの詳細ページのタブがあります。ビューアは、開発者が設定した属性を環境ファイル中に「属性=値」という形でマップし、ブローカ用の環境ファイルを作成します。もし同じ名前の環境ファイルが存在していたら、ビューアはその環境ファイルを上書きしてしまいます。

「General」詳細ページでは、表 6.2 に示すフィールドを埋めてください。

表 6.2 : 新規ブローカの「General」詳細フィールドの値

| フィールド | 値 |
|---|---|
| Host | ブローカを起動したいマシン名を指定します。 |
| Port | 8800。現在使われていないポート番号を指定します。このチュートリアルではブローカは 7800 を使用します。 |
| Start Directory | ブローカを起動したいディレクトリを指定します。 |
| Env File Path | 環境ファイルを作成するディレクトリ。 |

起動コマンドには、既にデフォルトの値が記入されています。今回は変更しませんが、変更する場合には「Custom Command Line」チェックボックスをクリックすると編集可能になります。

ウィンドウの一番上の File タブをクリックして、「File」詳細ページに移動します。ログファイル、エラーファイルはデフォルトで「broker.log」「broker.err」と指定されていますが、変更したい場合には編集できます。今回は編集しません。

DEBUG Level ボックスでは Error フィールドの横の下向きの矢印をクリックし、値を ERROR から DEBUG に変更します。

OK をクリックします。Add Service ウィンドウが消え、図に示すようにサービスリストウィンドウには、新規のブローカエントリが現れます。

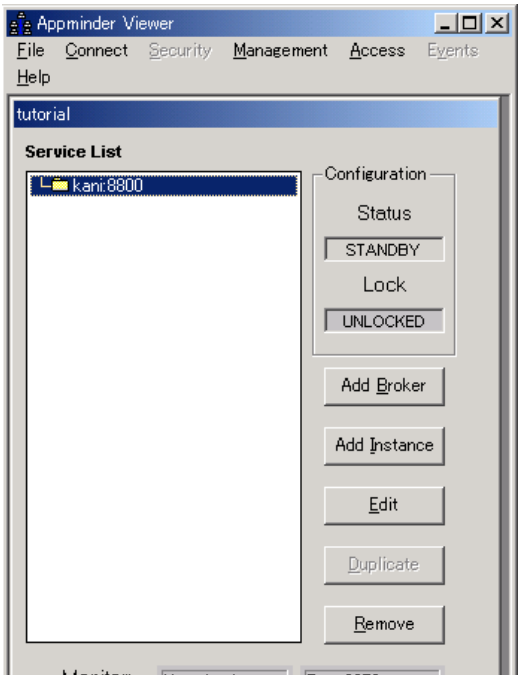


図 6.8：ブローカ追加後のサービスリスト

## サーバプロセスの追加

次に、この設定にブローカに登録するサーバプロセスを追加していきます。「Add Instance」ボタンをクリックします。Add Instance ウィンドウの「General」詳細ページが Service List ウィンドウの右側に表示されます。

ビューアは、ブローカを追加したときに指定した Host と同じ値を自動的に表示します。

Port フィールドは空のままにしておきます。オペレーティングシステムがサーバプロセスのポート番号を割り当てるからです。

次のフィールドを埋めます。( また起動コマンドは、絶対パスを指定するか、.\を最初に付けてください。)

- Executable Name：basics

この後「Interface Name」と「Env File Name」フィールドをクリックすると、「Executable Name」フィールドで指定した「basics」は反映され、それぞれ「basics」，「basics.env」がデフォルトとして表示されます。また「Command Line」にも起動コマンドが表示されます。

次に起動ディレクトリを指定します。実行オブジェクト「basics」があるディレクトリを記入してください。この後「Env File Path」をクリックするとデフォルトで起動ディレクトリと同じディレクトリが表示されます。

ウィンドウの一番上の File タブをクリックして、「File」詳細ページに移動します。ログファイル、エラーファイルはデフォルトで「basics.log」「basics.err」と指定されていますが、変更したい場合には編集できます。今回は編集しません。

DEBUG Level ボックスでは、Error フィールドの横の下向きの矢印をクリックし、値を ERROR から DEBUG に変更します。

OK をクリックします。Add Instance ウィンドウが消え、図 6.9 に示すようにサービスリストウィンドウには basics サーバプロセスが現れます。

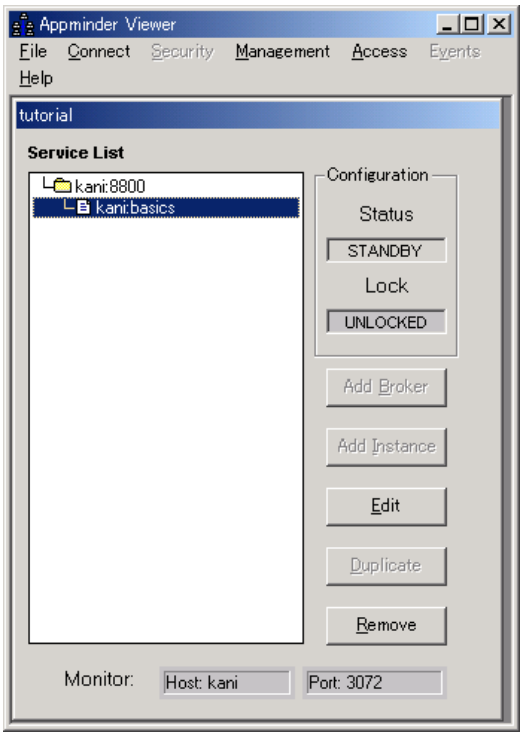


図 6.9：サーバプロセス追加後のサービスリスト

## サブブローカの追加

設定にサブブローカを追加するには、Service List ウィンドウでブローカエントリをハイライトしてから「Add Service」ボタンをクリックします。 Add Service ウィンドウの「General」詳細ページが Service List ウィンドウの右側に表示されます。

次のフィールドを埋めてください。

- Port：9800。利用可能なポート番号を指定します。モニタブローカは 7800、親ブローカは 8800 を使用しています。Port フィールドを埋めたら、ビューアは自動的に-p 9800 オプションを追加します。

File タブをクリックして「File」詳細ページに移動します。

DEBUG Level ボックスでは、Error フィールドの横の下向きの矢印をクリックし、値を ERROR から DEBUG に変更します。

OK をクリックします。Add Service ウィンドウが消え、サービスリストウィンドウにはサブブローカエントリが現れます。

### 設定の保存

これでブローカ、サブブローカ、basics サーバのプロセスが 1 つ入った tutorial 設定ができました。ここで一旦これまでの作業を保存します。File プルダウンメニューから「Save」オプションを選択します。これにより tutorial.cfg ファイルが New ダイアログ・ボックスで指定したディレクトリに書き込まれます。

## 設定をアクティブにする

tutorial 設定のブローカ、サブブローカ、サーバプロセスを起動するには Management プルダウンメニューから「Begin Managing」オプションを選択します。ビューアはサービスリストウィンドウの右にアクティブウィンドウを表示します。

アクティブウィンドウのグリッドは、設定の中の各オブジェクトのステータス情報を表示します。エージェントが各オブジェクトを起動するにつれて、グリッド中のステータスが New から STARTING、VERIFYING、PINGED に変わっていくのを確認できます。下の図はブローカ、サブブローカ、basics サーバプロセスがすべて起動・ピングされた後の、tutorial 設定のアクティブウィンドウを示します。

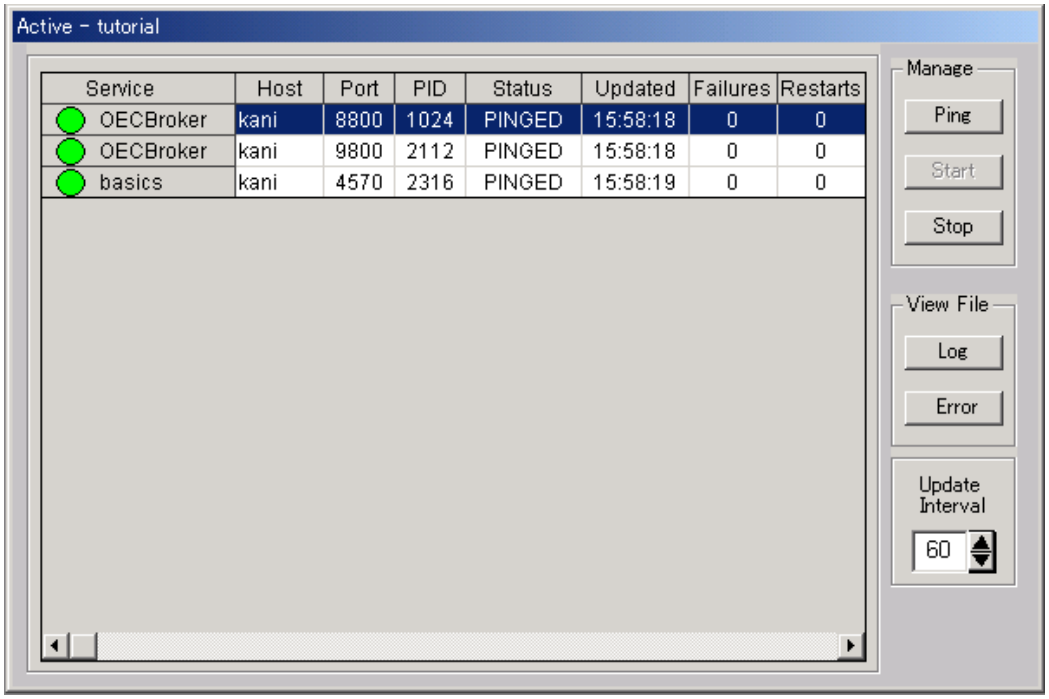


図 6.10：tutorial 設定のアクティブウィンドウ

デフォルトでは、ビューアは 60 秒ごとに現在のステータスを再表示します。Update Interval ボックスの上向き／下向き矢印をクリックして頻度を変更できます。

オブジェクトをピングしたり停止したりするには、グリッド中のオブジェクトを選択してからピングボタンや停止ボタンをクリックします。ブローカを停止すると、そのブローカに登録されているすべてのサーバプロセス、サブブローカ、そのサブブローカに登録されているすべてのサーバプロセスも、エージェントによって停止されてしまいますのでご注意ください。

## ログファイルの検証

オブジェクトのログファイルを検証するには、グリッド中のオブジェクトを選択してから View Log ボタンをクリックします。図 6.11 は、ブローカのログファイルを示しています。ログファイルには実行された RPC ファンクションとその事象についてのテキストメッセージがリストされます。

AppMinder がサーバを再起動するときには、AppMinder はそのサーバ用に新しくログファイルを作成します。その場合、ログファイル名は「`･･･.lo1`」「`･･･.lo2`」となります。このチュートリアルの例では「`basics.lo1`」「`basics.lo2`」となります。 サーバがデディケイテッド・サーバであれば、「`childxxx.log`」という命名規則になります。

ブローカの環境ファイル中 DCE_DEBUGLEVEL 属性を DEBUG, DEBUG に指定しているので、ビューアはログファイルにエラー、警告、コンテキストメッセージを書き込みます。左端の文字はメッセージのタイプを示します。

- D:コンテキストメッセージ
- E:エラーメッセージ
- W:警告メッセージ

ログファイルウィンドウの右端のスクロールバーを使ってログファイルをスクロールアップしたり、ダウンしたりすることができます。

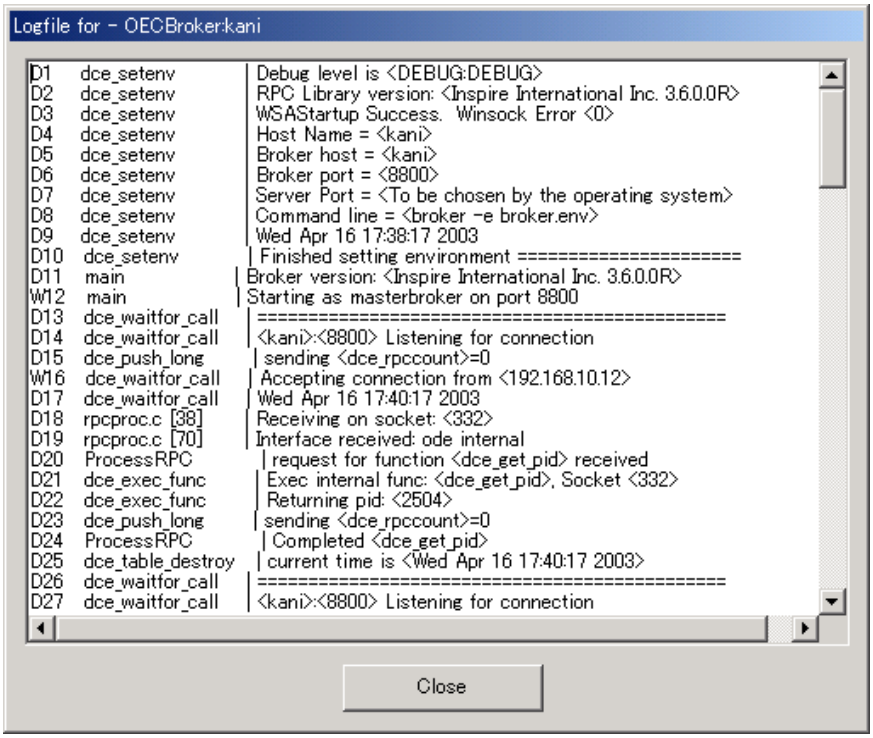


図 6.11：ログファイルウィンドウ

## サーバプロセスの複製

アクティブ設定に、ブローカやサーバプロセスを追加したり削除したりすることができます。この場合エージェントがオブジェクトの起動と停止をおこないます。basics サーバの複製プロセスを追加するには、サービスリストで basics サーバプロセスを選択し、「Duplicate」ボタンをクリックします。すると、複製サーバの環境ファイルとログファイルを指定するウィンドウが出てきます。各フィールドにはデフォルトで、オリジナルの basics サーバプロセスを設定したときに指定した値が入っていますが、フィールドの値を変更して、サーバプロセスごとに異なる環境ファイルやログファイルを指定することもできます。その後 OK ボタンをクリックしてください。

アクティブウィンドウには新しい basics エントリが現れ、エージェントはその basics サーバプロセスを起動します。下の図は「Duplicate」ボタンをクリックしたときに表示されます。

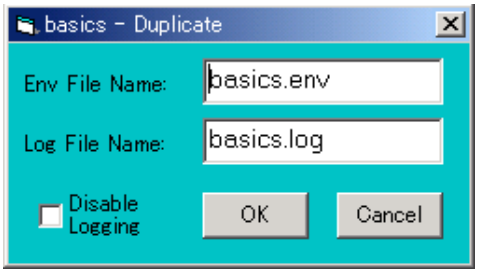


図 6.12 ： サーバプロセスの複製

# シャットダウン

ここでは、以下の手順を示します。

- 設定をスタンバイにする
- 設定をクローズする
- ビューアとモニタの接続を切る
- モニタ、モニタブローカ、エージェントを停止する

## 設定をスタンバイにする

tutorial 設定をスタンバイにするには、Service List ウィンドウを選択してから Management プルダウンメニューから「Stop Management」を選択します。ビューアはモニタを呼び出し、モニタはエージェントにブローカ、サブブローカ、basics サーバプロセスを停止するよう指示します。アクティブウィンドウは消え、サービスリストウィンドウにはスタンバイ状態になった設定が示されます。

## 設定をクローズする

設定をクローズするには、File プルダウンメニューから「Close」オプションを選択します。basics サーバプロセスを複製するなど、変更を加えていますので、ビューアからはクローズする前に設定を保存するかどうか聞かれます。「はい」を選択して、変更を保存してください。

## モニタから接続を切り離す

ビューアとモニタの接続を切り離すには、Connect プルダウンメニューから「Disconnect」を選択します。次に File プルダウンメニューから「Exit」を選んで、ビューアのセッションを終了します。モニタとの接続を切り離し、ビューアを終了しても、モニタとエージェントを停止したわけではありません。モニタとエージェントはまだ稼働しています。

## AppMinder コンポーネントの停止

モニタ、モニタブローカ、エージェントを停止するには、ソフトキルコマンドを発行します。

# 便利な機能

## Address Translation

Firewall 超えに関して、サーバマシンの IP アドレスが、クライアント PC から別の IP アドレスで見える場合（NAT 変換あり）の場合に有効な機能です。

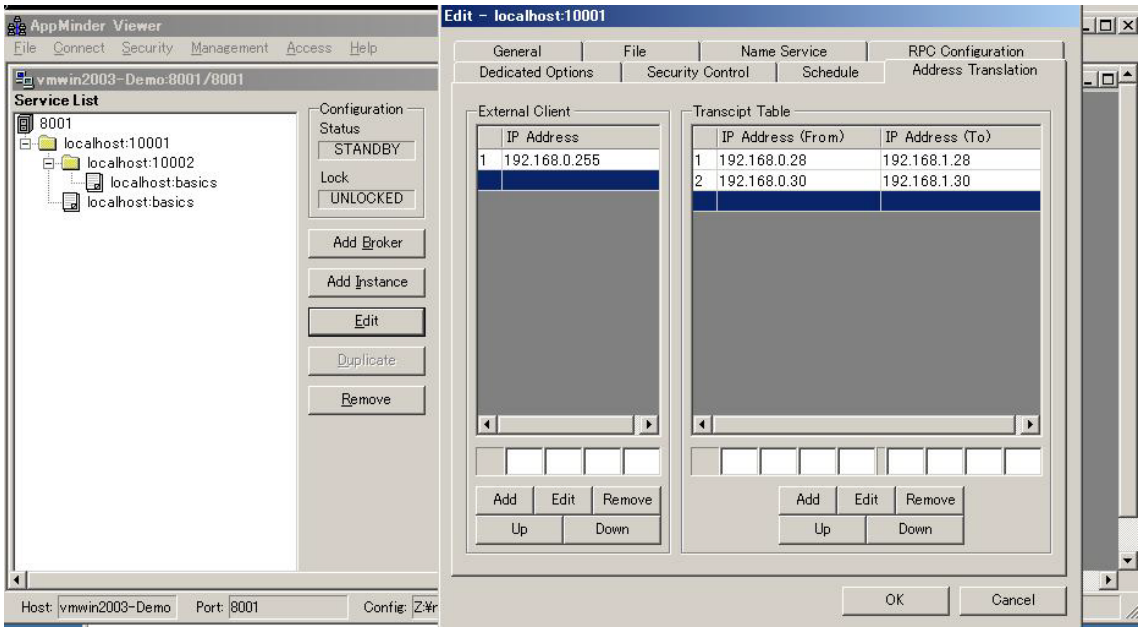


図 6.13 ： Address Translation タブ

図 6.13 は、localhost:10001 で起動する Broker の編集画面（Edit）にある、「Address Translation」タブになります。クライアントの IP アドレスが 192.168.0.255、つまり 192.168.0.1～192.168.0.254 の範囲にあるクライアントからのアクセスがあった場合、かつ Nextra サーバアプリケーションの IP アドレスが 192.168.0.28 であった場合、当該クライアントへのサーバ位置情報は、192.168.1.28 の IP アドレスに変換されます。Broker は、この変換された IP アドレスと Port 番号を、クライアントに送信し、クライアントは、これらの情報を元に Nextra サーバアプリケーションにアクセスすることになります。192.168.0.30 の IP アドレスで登録された Nextra サーバアプリケーションも同じように、Broker により、サーバ位置情報は、192.168.1.30 の IP アドレスに変換されクライアントに渡されます。

## サーバプロセスの起動 TCP/IP Port 番号の指定

AmViewer（ビューア）上で、「Edit」ボタンをクリックし"Use Explicit Port"をチェックして任意の Port 番号を入力してください。 Port 番号を指定したアプリケーション・サーバを AppMinder から起動すると、この TCP/IP Port を使用して起動します。

# ご注意

## 商標権に関する注意



## 著作権に関する注意



Nextra
# AppMinder ユーザガイド

| | |
|---|---|
| 2011 年　3 月　9 日 | エージェントのローカルファイル（appmagt.cdb） |
| 2008 年 10 月 10 日 | v5 2nd Edition |
| 2007 年　6 月 28 日 | amviewer [-h *monitor_host*] [-p *monitor_port*]、および便利な機能の追加 |
| 2007 年　4 月 17 日 | 第 3 版発行 |
| 2007 年　3 月　1 日 | appmagt パフォーマンス改善につき、追加属性 |
| 2006 年 11 月 13 日 | appmagt に、-pname <pipe name> オプションの追加 |
| 2004 年　9 月　1 日 | AMAGENT_MGMTINTERVAL の変更（40->120） |
| 2003 年 12 月 26 日 | 第 2 版発行 |
| 2003 年　5 月 25 日 | 「プロセスのステータスシンボル」追加 |
| 2003 年　4 月 18 日 | 初版発行 |

著者　Inspire International Inc.