基礎から覚える ADV+++

著：アライコウ

『はじめに』

ノベル・アドベンチャーゲーム制作ツールは確認できるだけでも、20 種類近くが個人プログラマーや企業によってリリースされています。

私はアマチュアクリエイター時代から、さまざまなツールを使ってきました。いずれも個性があり高機能で、最近では iPhone や Android といったスマートフォン向けにもリリースできるツールが珍しくありません。

ADV+++も、そんなツールのひとつです。ただし他のツールと明確に違う点がひとつあります。プロフェッショナル仕様と銘打ち、商用で使用されているシステムがそのまま公開されているのです。アマチュアクリエイターはもちろん、プロのクリエイターにとって大変に使いがいのあるツールなのです。

私もこのツールを用い、商用作品をリリースすることができました。しかし今までに触れてきたツールと比べてだいぶ感触が異なり、また他に解説しているサイトが見られなかったため、これから覚えてみたいという人にとって、習得するまでにある程度時間がかかると思いました。

その時間を短縮する助けになりたいというのが、本書を執筆したきっかけです。できることの範囲が非常に広いツールなので、タイトルどおりに基礎的なことに重点を置いて解説することにしました。そして、それらの基礎テクニックだけで充分に遊べる作品になります。

ADV+++は一度コツを覚えれば、次々にいろいろな機能を試したくなる魅力を備えています。ですので、まずは一通りのことを覚えてみましょう。本書が ADV+++ユーザーの増加、ひいては素晴らしい作品が誕生するきっかけになることを願っています。

『目次』

『はじめに』 2

第 1 章　ADV+++とは？ 6
『クロスプラットフォームでリリース』 7
『ADV+++の機能』 9

第 2 章　制作準備をしよう 13
『ファイル・フォルダ構成の把握』 14
『扱えるファイルの種類』 16
『シナリオファイルの書き方』 18
『スクリプトライブラリー』 21
『リストファイルに登録する』 24
『タイトル画面までの流れ』 26
『デバッグ情報を見る』 28

第 3 章　スクリプトを記述しよう 29
『スクリプトファイルを読み込む』 30
『テキストを表示する』 35
『テキストを装飾する』 44
『メッセージ送りアイコンの表示方法』 52
『タスクを生成する』 54
『背景画像を表示する』 56
『キャラクター画像を表示する』 64
『名前ウィンドウを表示する』 69
『顔ウィンドウを表示する』 71
『BGM を再生する』 74
『効果音を再生する』 77

『ボイスを再生する』 79
『ムービーを再生する』 80
『ウェイトを挿入する』 83
『画面を揺らす』 84
『選択肢とジャンプ』 87
『変数と条件分岐』 92
『画像に変化を加える』 102
『テキストをレイヤーにロードする』 112
『サブルーチンを使用する』 115
『テキスト表示中のアクション』 118
『クリッカブルマップ』 120
『ブラウザを起動する』 124
『セーブ関連機能を使用する』 125
『チャプター選択モードを使用する』 127
『マップモードを使用する』 131
『その他のモードを使用する』 135
『環境エフェクトを使用する』 136
『アチーブメントを使用する』 139
『外部フォントを使用する』 140
『スクリプトライブラリーに追加する』 143

第 4 章　システムを変更しよう 147
『00_common - 共通』 148
『00_strings - 文字列』 157
『01_title - タイトル』 159
『02_game - ゲーム本編』 167
『03_saveload - セーブ・ロード』 172
『04_options - オプション』 178
『05_cg - CG 鑑賞』 184
『06_music - 音楽鑑賞』 192

『07_scene - シーン鑑賞』 198
『09_home - ホームメニュー』 205
『11_chapter - チャプター選択』 207
『12_map - マップ選択』 210
『13_manual - 操作説明』 215
『achievements - アチーブメント』 217

第５章　作品をリリースしよう 219
『ダウンロード形式でリリース』 220
『メディアにパッケージしてリリース』 221
『iOS 向けにリリース』 227
『Android 向けにリリース』 231

第１章　ADV+++とは？

『クロスプラットフォームでリリース』

●Windows、Mac、iOS、Android に対応

ADV+++は商用で採用されているエンジンを、そのまま一般向けに使えるようにしたノベル・アドベンチャーゲーム制作ツールです。Windows をはじめ、Mac、モバイル環境の iOS、Android にもリリースできるクロスプラットフォーム仕様となっています。

モバイル端末は画面の解像度がそれぞれの機種によって違いますが、サイズを自動的に調節して表示できるので、パソコン向けに制作したものをモバイル向けに移植するということも非常に容易となっています。

ADV+++にはいくつかの種類があり、用途によって名称が違います。

【ADV+++ for PC】

Windows 上で動作する制作環境。画面サイズが 16:9 の「ADV+++ ｜ HD」がある。

モバイル向けの制作環境である「ADV+++ ｜ Mobile」も含み、横画面の「ADV+++ ｜ Mobile_H」と縦画面の「ADV+++ ｜ Mobile_V」に分類される。

対応 OS：Windows 10 / 8.1 / 7

【ADV+++ for MAC】

Mac 上で動作する制作環境。「ADV+++ ｜ HD」で制作したデータをそのまま流用して使える。専用アプリのテスト環境。

対応 OS：macOS 10.10（Yosemite）以降

【ADV+++ for iOS】

iOS 端末で動作する実行環境。専用アプリのテスト環境。

対応 OS：iOS 9.0 以降

【ADV+++ for Android】

Android 端末で動作する実行環境。専用アプリのテスト環境。

対応 OS：Android OS 4.0.3 以降

Windows と Mac は制作環境と実行環境が同じになりますが、iOS と Android はそうではありません。「ADV+++ for iOS」と「ADV+++ for Android」はいわばゲーム機で、このアプリ上で「ADV+++ | Mobile」で作られたゲームを動作させるのです。

ただし「ADV+++ for iOS」は規約の関係で App Store ではリリースされず、後述するデベロッパーライセンスを持つ人だけが入手できます。主にテストプレイ用に使用することになります。

「ADV+++ for Android」も本書執筆時点では、デベロッパーライセンスを持つ人だけが入手できます。

●32-bit 版と 64-bit 版

ADV+++は 32-bit 版と 64-bit 版があります。32-bit の OS 上では 32-bit 版のみを実行でき、64-bit の OS では 32-bit 版と 64-bit 版の両方を実行できます。64-bit 版のほうが実行時の動作パフォーマンスが向上しますので、可能であればこちらを使いましょう。

●デベロッパーライセンスを購入できる

ADV+++は全機能を無料で使用できますが、デベロッパーライセンスを購入すれば開発者コミュニティーに参加できます。ここでは開発版（Developer Preview）が随時リリースされますので、いち早く新機能を試すことが可能です。ただし開発版を使用したゲームの公開・配布・販売は禁止されています。正式版リリースのスケジュールには相談に乗ってもらえます。

また、要望への最優先対応なども受けられますが、必ず応えてくれるとは限りません。そして機能の追加・拡張などは、制作されているゲームで実際に使用されることが前提になります。

『ADV+++の機能』

●チュートリアルで機能を試す

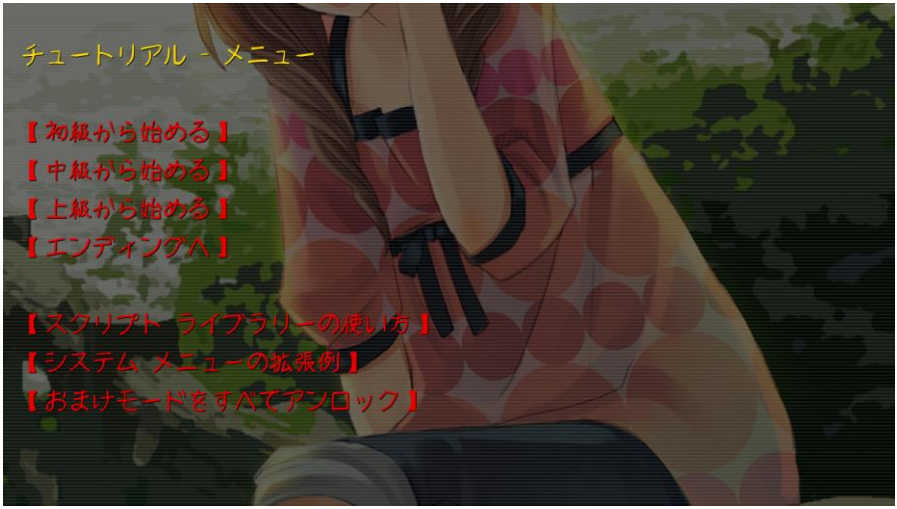
さっそく ADV+++の開発環境を公式サイトからダウンロードしましょう。

【ADV+++公式サイト】

http://www.yox-project.com/jp/adv_ppp/index.htm

ダウンロードページに前述のとおりいくつかの種類がありますが、本書では HD 版を使用するという前提で進めていきます。

ファイル、フォルダの種類に関しては次章で解説しますので、まずは adv_ppp_d.exe（64-bit の OS なら adv_ppp_x64_d.exe でも可）を実行しましょう。このファイルを実行すると、最初にオートビルドが行われます。オートビルドとはデータを ADV+++で扱える形式に変換する作業です。これが完了するとタイトル画面が表示されますので、「NEW GAME」を選択しましょう。



チュートリアルメニューという画面になりました。初級、中級、上級と分かれており、どういったテキスト表示や画面表示、演出が可能なのかということを試せます。

チュートリアルのスクリプトの内容はヘルプファイルの「ADV+++」→「ゲームの制作」→「3.チュートリアル」で見ることができますので、本書と平行して見るようにするとよいでしょう。そしてチュートリアルをすべて終えると、CG 鑑賞、音楽鑑賞、シーン鑑賞が開放されます。

●多彩で高機能なシステム

ADV+++は非常に高機能なシステムを備えており、制作者が一から構築することなく使えるという特徴があります。

詳しくは第 4 章で解説しますが「base¥config」フォルダにあるファイルで、それらのシステムを設定します。設定項目には次のようなものがあります。

00_common	ゲーム全般の基本的設定
00_strings_**	システムメッセージ
01_title	タイトル画面
02_game	メッセージログ等ゲーム本編のシステム
03_saveload	セーブ・ロード画面
04_options	オプション画面
05_cg	CG 鑑賞画面
06_music	音楽鑑賞画面
07_scene	シーン鑑賞画面
09_home	おまけモード等に使うホーム画面
10_nameentry	名前入力画面
11_chapter	チャプター選択画面
12_map	マップ選択画面
13_manual	操作方法などの表示
14_login	ログインモード
achievements	ゲーム達成度の管理

※08_は現在未使用。

たとえばセーブ・ロード画面はサムネイル、シーン名、プレイ時間、セーブ日時を表示する機能が最初からあります。オプション（コンフィグという表記もよく使われる）ではメッセージ速度やサウンドのボリュームをスムーズに切り替えられます。

名前入力画面は現在機能を改訂中、ログインモードは商業向けの機能なので、本書では取り上げないことにします。

●簡易なスクリプト

ノベル・アドベンチャーゲーム制作ツールにおいては、スクリプト記述式が一般的です。ADV+++のスクリプトは比較的シンプルな設計となっています。

```
sPlayS
        SID_S0, "sound.dat", FID_SOUND_SYS_S_DECIDE, NOLOOP
```

これは効果音を鳴らすときのスクリプトの例です。より詳しい解説は次章でしますが､命令名とパラメーターを記述するだけで済むという特徴があります。

第２章　制作準備をしよう

『ファイル・フォルダ構成の把握』

●ADV+++の構成要素

ADV+++でゲームを作るにあたり必要なフォルダ・ファイル構成は、以下のようになっています。

実行ファイル	
adv_ppp.exe adv_ppp_d.exe adv_ppp_x64.exe adv_ppp_x64_d.exe	「_d」とついているものがデバッグ版で、「_x64」とついているものが 64-bit 版です。リリース版は開発段階では使用しません。
base フォルダ	
clean.bat clean_all.bat	実行するとオートビルドで変換したデータが削除されます。※「clean_all.bat」はセーブデータも削除。
develop.txt	開発環境の設定ファイルです。
config フォルダ	システム設定ファイルが入っています。
document フォルダ	ゲームに関連するドキュメントが入っています。
font フォルダ	フォントデータが入っています。
graphic フォルダ	画像ファイルが入っています。
movie フォルダ	動画ファイルが入っています。
music フォルダ	音楽ファイルが入っています。
script フォルダ	スクリプトファイルが入っています。
setup フォルダ	リリース版のセットアップ時に使用される画像ファイルが入っています。制作段階では使用しません。
sound フォルダ	効果音ファイルが入っています。
ui フォルダ	ユーザーインターフェイス用の画像ファイルが入っています。
voice フォルダ	音声ファイルが入っています。

●ファイル・フォルダ名について

ファイルとフォルダの名称は、開発環境のダウンロード時からあるものについては、基本的には編集しないで使用します。これらはリストファイルで管理されているので、名称を編集すると正常に動作しなくなるのです。

リストファイルについては後ほど解説します。

『扱えるファイルの種類』

●テキストファイルの扱い

テキストファイルのフォーマットは、一般的な txt 形式です。

●素材ファイルの扱い

素材ファイルのフォーマットは、以下のとおりです。

素材の種類	フォーマット
グラフィック	BMP / JPG
サウンド	WAV / OGG
ムービー	WMV / MP4

グラフィックは基本的には bmp 形式を使用します。jpg 形式は背景画像の容量をギリギリまで削減したいという場合に使用します。

キャラクター画像はαチャンネルを持つ 32-bit の bmp 画像になりますが、手持ちのグラフィック編集ソフトがこれを出力できないなら、「Picture Viewer PV32 / PV64」（以下、Picture Viewer と略す）という画像ビューアをダウンロードしてください。YOX-Project が無料で公開している画像ビューアで、これが ADV+++でのゲーム制作にとても役立ちます。

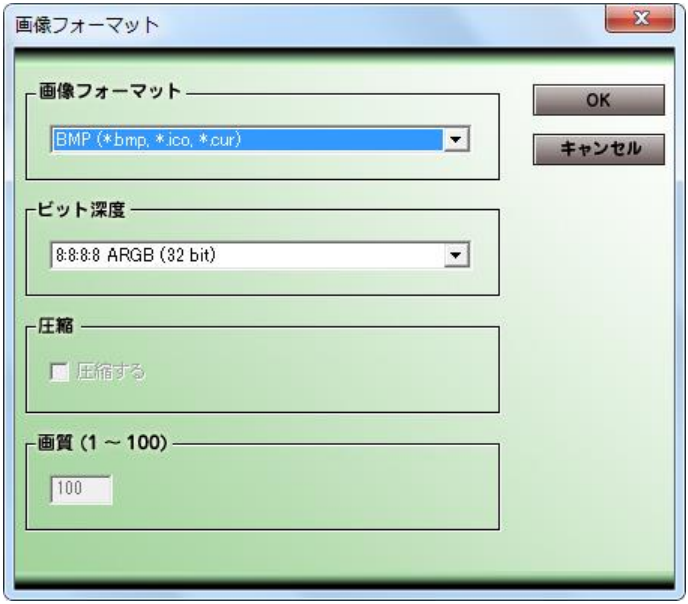
【Picture Viewer】

http://www.yox-project.com/jp/pv/index.htm

特徴は第一に、非常に高速であること。どれほど大きなサイズの画像でもすぐさま表示できます。psd ファイルも表示できるのが嬉しいところ。

そしてαチャンネルを持つ 32bit のビットマップ画像を表示、出力できるのです。その他、画像解像度やキャンパスサイズの変更など、機能は多岐にわたります。

では、32bit のビットマップ画像の出力方法を覚えましょう。α チャンネルを持つ png 形式の画像を用意して、読み込ませてください。そうしたらメニューの「ファイル」→「別名で保存」を選択して、形式は bmp にします。



ビット深度という項目があります。32bit となっているはずですから、そのまま保存してください。これで出力が完了します。

『シナリオファイルの書き方』

●スクリプトの書式

チュートリアルのファイルは「script¥tuto」の中にあります。この中から 0_main.txt を開いてみましょう。

第１章でも軽く触れましたが、命令名とパラメーターの組み合わせによってひとつのスクリプトが構成されます。

```
sPlayS
		SID_S0, "sound.dat", FID_SOUND_SYS_S_DECIDE, NOLOOP
```

一行目の『sPlayS』が命令名で、二行目の『SID_S0』『"sound.dat"』『FID_SOUND_SYS_S_DECIDE』『NOLOOP』がパラメーターです。二行目の頭には必ずタブで空白を挿入し、パラメーターはコンマ（,）で区切ります。

パラメーターはそれぞれ、以下のような型に分類されます。

型	略称表記	概要
string	str	文字列。『"テスト"』 のように ダブルクォーテーションで囲って記述します。
immediate	imm	即値（プログラムに記述した値。整数やその他の値)。text を除けばもっとも頻繁に使用する型です。
register	reg	レジスタ。変数の値を格納するのに使用します。
parameter	param	パラメーター。マクロ専用の型で、通常のスクリプトの記述には使用しません。
label	label	ラベル。ジャンプ先を指定するのに使用します。

text	text	テキスト。会話文、地の文などシナリオの記述に使用する命令『msg』固有の型です。

上記の例では『SID_S0』『FID_SOUND_SYS_S_DECIDE』『NOLOOP』が imm で、『"sound.dat"』が str になります。これらの型はヘルプファイルの命令解説を読む上で必要なので、覚えておいてください。

●ファイル・フォルダ名について

各ファイルを見てみればわかりますが、ファイル名は小文字です。ただしスクリプトに記述する際は大文字になっています。そしてアンダーバー（_）で階層を区切ります。拡張子は記述する必要はありません。

・sound¥sys¥s_decide.wav（実際のフォルダ階層とファイル名）
・FID_SOUND_SYS_S_DECIDE（スクリプトに記述する場合）

頭についている「FID」はファイル ID という意味です。この書式を守らないとエラーになってしまいます。

●コメント機能

ファイル内に「//」という記述が見られますが、これはプログラミング言語でよく使用されるコメント機能です。

```
// 環境の初期化（ノベル）

// 画面全体を暗くする

// タイトル
```

行頭に「//」と記述すれば、記述以降のテキストやスクリプトはゲーム上に反映されません。ファイルを読みやすくするためのメモや仕切り線に活用しましょう。

また「/*～*/」で囲めば、複数行をコメントアウトすることもできます。

```
/*
このメッセージは
ゲーム上には表示されません。
*/
```

『スクリプトライブラリー』

●効率よく制作するためのスクリプト集

script フォルダ内に lib というフォルダがあります。ここには頻繁に使用するであろう機能を使いやすくまとめたスクリプト集が入っており、スクリプトライブラリーと呼びます。

これをシナリオ内で呼び出し、状況に応じて値を変えるだけで効率よく使うことができます。

```
scriptSetParam7
        GID_CHAR1, FID_GRAPHIC_C_???, CHAR_X_C, CHAR_Y, 0,
0, 0
script
        "", FID_SCRIPT_LIB_CHAR_IN
```

これはキャラクター表示の例です。呼び出し元である char_in.txt を開いてみると、かなり大量のスクリプトが記述されていますが、これがたったの 4 行にまとめられているわけです。ADV+++の基礎とはスクリプトライブラリーをマスターすることにあると言っても過言ではありません。

●ライブラリーの種類

スクリプトライブラリーに入っているファイルは以下の表のとおりです。

ADV 用汎用ルーチン	
環境の初期化	adv_init.txt
環境の後始末	adv_uninit.txt
メッセージ表示前の処理	adv_msg_in.txt
メッセージ表示後の処理	adv_msg_out.txt

ノベル用汎用ルーチン	
環境の初期化	novel_init.txt
環境の後始末	novel_uninit.txt
メッセージ表示前の処理	novel_msg_in.txt
メッセージ表示後の処理	novel_msg_out.txt
背景	
ロード	bg_load.txt
モザイク切り替え	bg_change_mosaic.txt
ワイプ切り替え	bg_change_wipe.txt
連番ファイル	bg_movie.txt
連番ファイル（非同期）	bg_movie_async.txt
画面揺らし	bg_shake_m.txt
カメラ	
リセット	cam_reset.txt
キャラクター	
イン	char_in.txt
イン（非同期）	char_in_async.txt
アウト	char_out.txt
アウト（非同期）	char_out_async.txt
全アンロード	char_unload_all.txt
ローリング	char_rolling.txt
顔	
イン	face_in.txt
アウト	face_out.txt
エフェクト	
フィルター イン	vfx_filter_in.txt
フィルター アウト	vfx_filter_in.txt
ホワイトフラッシュ	vfx_flash_white.txt
画面揺らし	vfx_quake_drop.txt

開発	
背景チェッカー	dev_bg_checker.txt

※bg_shake_m.txt と vfx_quake_drop.txt は同一の処理

新規作成、または既存のファイルをアレンジして、オリジナルのスクリプトを追加することもできます。

『リストファイルに登録する』

●使用するファイルを明記

各スクリプトや素材ファイルは、リストに登録することで初めて使用できるようになります。例として、script フォルダ内の script.lst をテキストエディタで開いてください。

```
//#S:1
//#D:1
//#C:9
// ------------------------------------------------------------
"bin/main_logo_a.sce",
"bin/main_logo_b.sce",
"bin/main_game_update.sce",
"bin/main_opening.sce",
"bin/main_title.sce",
"bin/main_game.sce",
"bin/main_ending.sce",

// 汎用ルーチン ------------------------------------------------
// ADV 用
"bin/lib/adv_init.sce",
"bin/lib/adv_msg_out.sce",
"bin/lib/adv_msg_in.sce",
"bin/lib/adv_uninit.sce",

（以下略）
```

このような書式で、すべてのファイルが登録されています。登録されていないファイルをスクリプトで呼び出そうとすると、エラーが出るという仕様です。

ファイルの下のほうに「※ 以下にファイルを追加してください。」とあり、そこに作品に使用する各種ファイルを新規登録していくことになります。

●不要なものは削除

やっぱりこのファイルは不要、となったらリストから削除していきます。

特にリリース段階になったらチュートリアル関連のファイルは不要になるので、graphic、script 両フォルダにある tuto フォルダなどとともに削除するようにしてください。いろんなテキストファイルにチュートリアル関連の記述があるので、「tuto」でグローバル検索し、削除漏れがないようにしましょう。

『タイトル画面までの流れ』

●ロゴとオープニングを表示

ADV+++を起動すると、ロゴとオープニングが挿入されます。これらは独立したスクリプトファイルによって動作しています。

ファイル名	概要
main_logo_a.txt main_logo_b.txt	ロゴ表示画面用
main_opening.txt	オープニング用
main_title.txt	タイトル画面用
main_game.txt	ゲーム本編用

main_logo_a.txt、main_logo_b.txt に関わっているサークルや企業のロゴを表示します。実際は 1 つのファイルに複数のロゴを表示するスクリプトを記述できるので、上限が 2 つまでということはありません。

main_opening.txt はロゴ表示画面とタイトル画面の間で実行されます。ムービーなどがあれば、ここに挿入するのがよいでしょう。オープニングが不要の場合、オンとオフを容易に切り替えることができます。config フォルダの 00_common.txt を開いてください。

```
// オープニングの有無（0:なし, 1:あり, 2:ループ時のみあり）　　戻るまでの時間（秒）
1　　60
```

最初の値を『0』と書き換えれば、オープニングはスキップされます。

main_title.txt は、自分でタイトル画面を構築したいときに使用します。これを制御するのは同じく config フォルダの 01_title.txt です。

```
// 描画にスクリプトを使用するか？（0:使用しない, 1:使用する）
0
```

値を『1』に変更すると、キャラクターのロードと雪が降る演出がサンプルとして表示されます。このように動きのあるタイトル画面を作りたい場合に有用な機能です。

そしてタイトル画面の「NEW GAME」ボタンを経て main_game.txt が実行され、ゲーム本編が始まります。このファイルの編集からゲーム制作は開始します。

『デバッグ情報を見る』

●制作に不可欠な「DebugView」

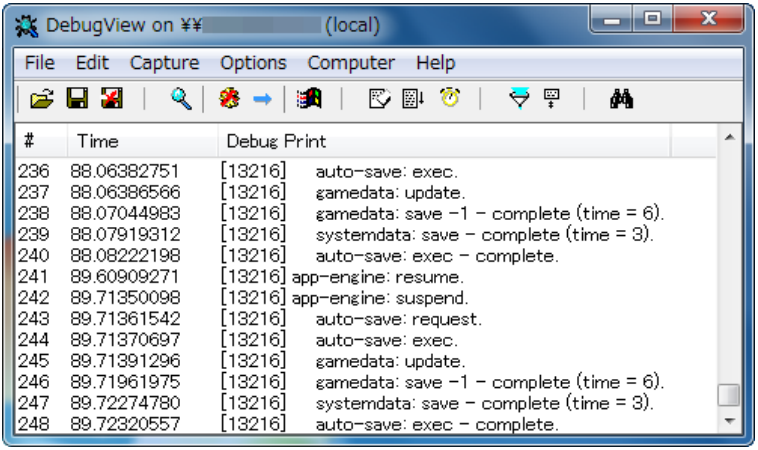
ADV+++はデバッグ機能も充実しています。デバッグ情報を見るには、Microsoft 純正のフリーウェアである「DebugView」をダウンロードします。

【DebugView】

http://technet.microsoft.com/ja-jp/sysinternals/bb896647

ADV+++を起動する前にこのアプリケーションを起動しましょう。エラーの内容などを把握することができます。これをデバッグログといいます。

今後、デバッグログを見る必要のある内容も扱いますので、できるだけADV+++と一緒に起動するようにしてください。



第3章　スクリプトを記述しよう

『スクリプトファイルを読み込む』

●ADV+++で最重要の命令

前章で解説したとおり、main_game.txt からゲーム本編が始まります。このファイルの中身がどうなっているかを見てみましょう。最初に覚えるべきところは以下の部分です。

```
// (※ 以下にゲーム本編のスクリプトを記述してください。)
// チュートリアル - メニュー
script
        "", FID_SCRIPT_TUTO_0_MAIN
```

スクリプトファイルを読み込む命令[script]です。オートビルドによって生成された script.dat の中から、「script¥tuto¥0_main.sce」のファイルを読み込むという意味です。これでチュートリアルが始まるようになっています。

ゲームはここから始まるという点で、[script]は ADV+++で最重要の命令と言ってもよいでしょう。また、こうやってスクリプトファイルを読み込ませるのが main_game.txt の役割とも言えます。スクリプトファイルを読み込まず、main_game.txt に直接ゲーム本編のスクリプトを記述していくこともできますが、作りやすさという観点からおすすめはできません。

さて、自分で作ったシナリオファイルを読み込ませるには、これを書き換えればよいのです。例として test フォルダとテキストファイル 00.txt を新規作成し、リストに登録します。

```
// (※ 以下にファイルを追加してください。)

"bin/test/00.sce",
```

登録するのは txt ではなく、オートビルド後に生成される sce ファイルであることに注意してください。あとは[script]を書き換えるだけです。

```
// テスト
script
        "script.dat", FID_SCRIPT_TEST_00
```

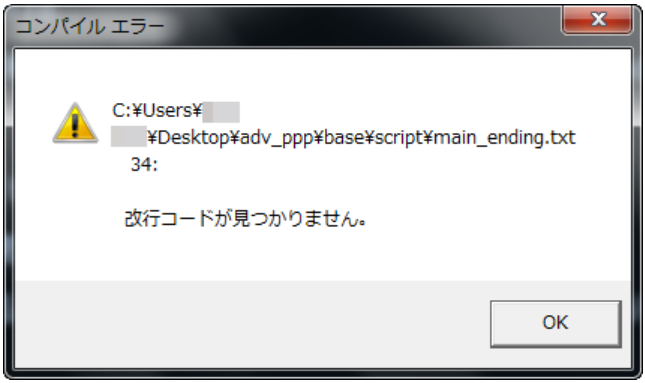
00.txt に何も記述していない状態だと、始まった直後に main_ending.txt で記述されたエンディングが始まってしまいます。main_ending.txt はゲーム本編の終了後に実行されるスクリプトなのですが、毎回実行されるのは面倒です。中身をごっそり削除してしまいましょう。

```
// include ----------------------------------------------------
// system
#include "sys/platform.txt"
#include "sys/system_macro.txt"
#include "sys/system_define.txt"
#include "sys/game_define.txt"
#include "sys/fid_font.txt"
#include "sys/fid_graphic.txt"
#include "sys/fid_music.txt"
#include "sys/fid_script.txt"
#include "sys/fid_sound.txt"

//////////////////////////////////////////////////////////////////////
// public:
//////////////////////////////////////////////////////////////////////
MAIN:
```

```
        // このスクリプトを終了する
        ret

//////////////////////////////////////////////////////////////////
// end of code
```

こんな風に、最低限の記述と区切りだけを残して削除するのです。[ret]はスクリプトを終了する命令で、すべてのスクリプトファイルの末尾に記述されています。

なお、最終行の「end of code」のあとに一行空いていますが、ここで改行しておかないとエラーが出てしまいますので、うっかり改行箇所を削除しないようにしてください。



●環境を初期化

ここから少しずつ 00.txt に肉付けをしていきます。まず必要なのが、環境を初期化するスクリプトです。環境の初期化とは、テキストのフォント関係やその他諸々を、最初に決定しておくことです。

前章で解説したスクリプトライブラリーに、初期化用のスクリプトファイル adv_init.txt と novel_init.txt があります。

```
script
        "", FID_SCRIPT_LIB_ADV_INIT
```

画面下部にウィンドウが出るアドベンチャータイプなら adv_init.sce を読み込みます。本書では基本的に、アドベンチャータイプを制作する前提で解説を進めていくことにします。

1 つめのパラメーターが『""』となっていますが、このように省略しても『"script.dat"』と指定したのと同じことになります。

```
script
        "", FID_SCRIPT_LIB_NOVEL_INIT
```

全画面にテキストを表示するノベルタイプなら novel_init.sce を読み込みます。

●環境の後始末

基本的にアドベンチャータイプだが、シーンによってはノベルタイプで見せる……そんな手法もあります。こういった切り替えをする際に、環境の後始末を行っておきましょう。

```
script
        "", FID_SCRIPT_LIB_ADV_INIT

script
        "", FID_SCRIPT_LIB_ADV_UNINIT

script
        "", FID_SCRIPT_LIB_NOVEL_INIT
```

```
script
        "", FID_SCRIPT_LIB_NOVEL_UNINIT
```

adv_uninit.txt と novel_uninit.txt の中身はどちらも同じで、全サウンドと全グラフィックを開放（消去）し、グラフィックの輝度を初期値に戻すというものです。ここでアドベンチャーモードが、ノベルモードが終わったという目印的なスクリプトとしても有用です。