WebSphere Liberty サーバーでの詳細なログ記録

WebSphere Liberty Profile (WLP) の `server` コマンドに関する「詳細なログオプション」についての質問に対応するために、`server` コマンドがどのように動作し、詳細度を制御する方法を明確にします。

WLP の `server` コマンドは、サーバーインスタンスを管理するために使用され、`start`、`stop`、`run`、`status` などのアクションが含まれます。例えば：

- `server start <serverName>` は、サーバーをバックグラウンドで起動し、出力をログファイル（例：`logs/console.log`）にリダイレクトします。
- `server run <serverName>` は、サーバーをフォアグラウンドで起動し、出力を直接ターミナルに表示します。

質問では、`server` コマンドの「詳細なログオプション」について尋ねられていますが、これはコマンド自体の出力の詳細度が増えることを意味するか、管理するサーバーの詳細なログを意味するかのどちらかです。オプションを探索した結果、`server` コマンドには `--verbose` や `-v` などの直接的なフラグはなく、詳細度はサーバーのログ動作に関連しています。

詳細なログの有効化

WLP では、ログの詳細度はサーバーのログ設定を通じて制御され、`server` コマンドのフラグを直接変更するのではなく、コマンドを呼び出す際に影響を受けます。以下に、詳細なログを有効にする方法を示します。

1. `server.xml` でのログ設定　詳細なログを有効にする主要な方法は、`server.xml` ファイル内の `<logging>` 要素を調整することです。このファイルは通常、`<WLP_HOME>/usr/servers/<serverName>/` にあります。詳細なトレース仕様を設定することで、ログの詳細度を増やすことができます。例えば：

```
<logging traceSpecification="*=all" />
```

- `*=all` はすべてのトレースポイントを有効にし、ログを非常に詳細にします（デバッグに便利）。
- よりターゲット指向の詳細度を設定するには、コンポーネントを指定することもできます。例えば、`*=info:com.example.*=debug` は、デフォルトのレベルを `info` に設定し、`com.example` パッケージのレベルを `debug` に設定します。

他にも有用な属性には以下があります：- `logLevel`：一般的なログレベルを設定します（例：`INFO`、`DEBUG`、`TRACE`）。- `consoleLogLevel`：`console.log` またはターミナルに書き込まれるメッセージのレベルを制御します（例：`DEBUG` または `TRACE`）。

より細かいコンソールレベルの例：

```
<logging consoleLogLevel="DEBUG" traceSpecification="*=audit" />
```

`server start` を実行すると、ログ（詳細な出力を含む）は `logs/console.log` に書き込まれます。`server run` を使用すると、この詳細な出力は直接ターミナルに表示されます。

2. 環境変数の使用　環境変数を使用してログの詳細度を制御することもできます。これらの変数は、server.xml の設定を上書きまたは補完します。例えば、server コマンドを実行する前に WLP_LOGGING_CONSOLE_LOGLEVEL 変数を設定します：

```
export WLP_LOGGING_CONSOLE_LOGLEVEL=DEBUG
server start <serverName>
```

- これにより、コンソールログレベルが DEBUG に設定され、より詳細な出力が得られます。
- 有効なレベルには INFO、DEBUG、FINE、FINER、FINEST などがあります。

server run の場合、これによりターミナル出力がより詳細になり、server start の場合は console.log に影響を与えます。

3. JVM 特有の詳細なオプション　特定の詳細な出力（例：ガベージコレクションログ）が必要な場合は、JVM オプションを追加することもできます（ただし、これはサーバーの実行時関連であり、server コマンド自体ではありません）。<WLP_HOME>/usr/servers/<serverName>/ にある jvm.options ファイルに -verbose:gc を追加するか、server.xml を通じて含めることができます：

```
<jvmEntries verbose="gc" />
```

これにより、GC 関連のログが生成され、通常は console.log または server run でターミナルに表示されます。

server コマンドとの関連性

- **直接的な --verbose オプションなし**：server コマンド自体（例：server start）には、実行ステップの詳細を表示するための --verbose や類似のフラグはありません。基本的なステータスメッセージ（例：“サーバーが開始されました”）を超えることはありません。
- **サーバーのログが詳細度を制御**：見える詳細度はサーバーのログ設定に依存します。server run を使用してターミナルでログをリアルタイムで観察するか、server start を実行してログファイルを確認します。

例えば：- server.xml に <logging traceSpecification="*=all"/> を含めると、server run <serverName> を実行するとターミナルに詳細なログが表示されます。- server start <serverName> を実行すると、同じ詳細度が logs/console.log に表示されます。

推奨事項

server コマンドを使用して詳細なログを得るためには：1. server.xml を編集して詳細な traceSpecification を含めます（例：最大の詳細度を得るために *=all または特定のレベル *=debug）。2. 代替として、環境で WLP_LOGGING_CONSOLE_LOGLEVEL=DEBUG を設定します。3. server run を使用して詳細な出力を直接確認するか、server start を実行して logs/console.log を確認します。

このアプローチにより、デバッグや監視の要件に応じた詳細なログを得ることができます。

もし、コマンドの実行詳細度（サーバーログではなく）についての具体的なものを意味していた場合は、追加の説明をお願いします！それ以外の場合、サーバーのログ設定を上記のように構成するのが、`server` コマンドで詳細なログを有効にする標準的な方法です。

注意：`logs` ディレクトリには、トレースファイルなどのログが含まれています。