ezSpectraAPI マニュアル（API バージョン 1.0.0）

主なクラス

OaktreeLab.ezSpectra.ezSpectraAPI

API 本体。

OaktreeLab.ezSpectra.Spectrum

スペクトルデータ。後述の Datum クラスの配列をラップしている。

OaktreeLab.ezSpectra.Spectrum.Datum

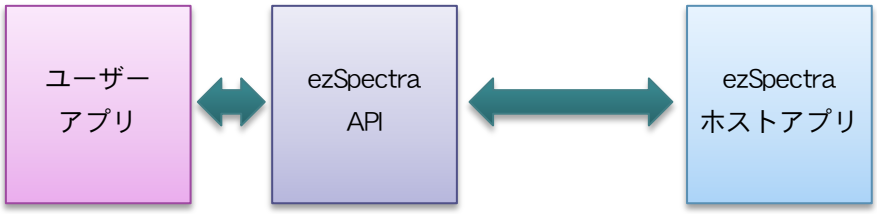
波長とスペクトル強度の対。

OaktreeLab.ezSpectra.AnalysisResult

スペクトル解析結果。

API の構造及び動作原理

API は、ezSpectra アプリ（ホストアプリ）の機能を、ユーザーアプリから利用する構造となります。



`Connect()`を実行するとホストアプリが自動的に起動して、`API` を通じて測定を行います。`Close()`を実行するとホストアプリとの接続を終了し、ホストアプリを閉じます。

ezSpectraAPI クラス

コンストラクタ

ezSpectraAPI()

ezSpectra API のインスタンスを作成する。

メソッド

void Connect()

ezSpectra ホストアプリを起動し接続する。接続が完了するまで待つ。

void Connect(bool wait)

ホストアプリを起動し接続する。

`wait` ： 接続が完了するまで待つか？

void Initialize()

ホストアプリとの接続状態を初期化する。`Connect( false )`を実行した場合のみ必要。

void Reset()

アイドル状態に戻す。すべての測定パラメータは初期化される。

void Close()
ホストアプリとの接続を終了する。KeepAppOpen が false の場合はホストアプリを自動的に終了する。

void Dispose()
リソースを破棄する。内部で Close() を呼び出す。

void Ping()
ホストアプリとの疎通確認。

void ExecuteSampling()
サンプリングを実行する。完了するまで待つ。

void ExecuteSampling(bool wait)
サンプリングを実行する。
wait　：完了するまで待つか？

void AbortSampling()
実行中のサンプリングを中断する。

void SetExposureTime(uint ExposureTime)
露光時間を設定する。設定が完了するまで待つ。
ExposureTime　：露光時間（ミリ秒単位）

void SetExposureTime(uint ExposureTime, bool wait)
露光時間を設定する。
ExposureTime　：露光時間（ミリ秒単位）
wait　：完了するまで待つか？

void SetAutoExposure(bool mode, bool wait)
自動露光を設定する。
mode　：自動露光を有効にするか？
wait　：自動露光を有効にする場合、安定するまで待つか？

void ExecuteDarkCurrentSampling()
ダーク補正を開始する。終了するまで待つ。

void ExecuteDarkCurrentSampling(bool wait)
ダーク補正を開始する。
wait　：終了するまで待つか？

void AbortDarkCurrentSampling()
実行中のダーク補正を中断する。

プロパティ

bool KeepAppOpen
Close() を実行したときにホストアプリを終了するか否かを取得/設定する。

int BlockingWaitLoopInterval
各メソッドを wait=true で実行したときの、終了待ちループの実行間隔をミリ秒単位で取得/設定する。

bool IsIdle
ホストアプリがアイドル状態（測定を開始できる状態）か否かを取得する。

SamplingQualityType SamplingQuality

サンプリング品質を取得/設定する

`SamplingQualityType.None` ： ノイズ低減無し

`SamplingQualityType.Low` ： ノイズ低減低

`SamplingQualityType.High` ： ノイズ低減高

bool IsHeld

サンプリングが完了したか否かを取得する。

uint ExposureTime

露光時間をミリ秒単位で取得/設定する。設定する場合は、設定が完了するまで待つ。

bool AutoExposure

自動露光の設定を取得/設定する。自動露光を有効にした場合、露光が安定するまで待つ。

bool IsAutoExposreStable

自動露光が安定しているか否かを取得する。

Spectrum CurrentSpectrum

現在の最高値が 1 に正規化されたスペクトルを取得する。

Spectrum CurrentRawSpectrum

現在の正規化されていないスペクトルを取得する。

AnalysisResult CurrentAnalysisResult

現在のスペクトル分析結果を取得する。

Spectrum クラス

コンストラクタ

Spectrum()

要素数 256 で初期化したインスタンスを作成する。

Spectrum(int size)

要素数を指定して初期化したインスタンスを作成する。

`size` ： 要素数

Spectrum(Spectrum s)

データをコピーして初期化したインスタンスを作成する。

`s` ： 元のスペクトル

Spectrum(double[] WaveLengthList)

波長を設定して初期化したインスタンスを作成する。

`WaveLengthList` ： 波長のリスト

Spectrum(double[] WaveLengthList, double[] ValueList)

波長とスペクトル強度で初期化したインスタンスを作成する。

`WaveLengthList` ： 波長のリスト

`ValueList` ： スペクトル強度のリスト

メソッド

void Normalize()
　スペクトル強度の最大値を 1 として正規化する。

IEnumerator<Spectrum.Datum> GetEnumerator()
　IEnumerator を返す

プロパティ

int Length
　スペクトルの要素数を取得する。

double[] WaveLengthList
　波長のリストを取得する。

Spectrum.Datum Peek
　スペクトルのピークをもつ要素を取得する。

インデクサ

Spectrum.Datum this[int index]
　特定の要素を取得・設定する。

演算子

double、double[]、Spectrum との四則演算を提供します。

double との演算の場合、スペクトルのすべての要素に対して演算を行い、結果の Spectrum 型を返します。
例えば、(Spectrum)s + (double)a は以下のように定義されます。

```
ans(i) = s(i) + a
```

double[]との演算の場合、スペクトルの各要素に対して、double[]の値と演算を行い、結果の Spectrum 型を返します。要素数は同じでなければなりません。
例えば、(Spectrum)s + (double[])a は以下のように定義されます。

```
ans(i) = s(i) + a(i)
```

Spectrum との演算の場合、スペクトルの各要素に対して演算を行い、結果の Spectrum 型を返します。要素数及び波長は同じでなければなりません。
例えば、(Spectrum)s1 + (Spectrum)s2 は以下のように定義されます。

```
ans(i) = s1(i) + s2(i)
```

キャスト

double[]
　スペクトル強度の配列を返す。

Spectrum.Datum クラス

コンストラクタ

Datum()
　波長とスペクトル強度を NaN で初期化したインスタンスを作成する。

Datum(double wavelength, double value)
　波長とスペクトル強度を指定してインスタンスを作成する。

Datum(Datum datum)

元の Datum と同じ内容のインスタンスを作成する。

プロパティ

double WaveLength

波長を取得/設定する。

double Value

スペクトル強度を波長を取得/設定する。

AnalysisResult クラス

プロパティ

double CCT

相関色温度[K]を取得する。

double x

xy 色度図における色座標 x を取得する。

double y

xy 色度図における色座標 y を取得する。

double u

uv 色度図における色座標 u を取得する。

double v

uv 色度図における色座標 v を取得する。

double[] Ri

各試験色の演色性評価数(i=1～15)を取得する。Ri[1]～Ri[15]にそれぞれの値が格納されている。

double Ra

平均演色性評価数を取得する。

double Lux

照度[Lux]を取得する。

double PPFD

光合成有効光量子束密度[μmol/m²s]を取得する。

プログラム例

最も基本的なサンプリング

自動露光ＯＮ、サンプリング品質を「高」に設定してサンプリングを行い、結果を取得します。

```
// ezSpectraAPI のインスタンスを作成する
ezSpectraAPI ezs = new ezSpectraAPI();

// ホストアプリを起動して接続する
ezs.Connect();

// サンプリング品質：高
ezs.SamplingQuality = SamplingQualityType.High;

// 自動露光 ON
ezs.AutoExposure = true;

// サンプリングを実行する
ezs.ExecuteSampling();

// サンプリング結果を取得する
Spectrum s = ezs.CurrentSpectrum;

// スペクトル解析結果を取得する
AnalysisResult ar = ezs.CurrentAnalysisResult;

// API の接続を閉じる
ezs.Close();
```

連続サンプリング

ExecuteSampling を繰り返し呼び出すことで、何度もサンプリングすることができます。

```
Spectrum[] s = new Spectrum()[10];
for ( int i = 0 ; i < 10 ; i ++ ) {
    // サンプリングを実行する
    ezs.ExecuteSampling();

    // サンプリング結果を取得する
    s[i] = ezs.CurrentSpectrum;
}
```

リアルタイムなスペクトルの取得

サンプリングの統計処理を行わずに、計測されたリアルタイムのスペクトルを取得することもできます。ただしこの場合、CurrentSpectrum で取得する間隔が露光時間より短い場合、前回と同じスペクトルを取得することに注意してください。

```
// ezSpectraAPI のインスタンスを作成する
ezSpectraAPI ezs = new ezSpectraAPI();
```

```
// ホストアプリを起動して接続する
ezs.Connect();

while ( true ) {
    // リアルタイムのスペクトルを取得する
    Spectrum s = ezs.CurrentSpectrum;

    // 1秒間休止する
    Thread.Sleep( 1000 );
}
```