人にやさしく、地球にやさしく
Thoughtfully, Living with the Earth
WELCOM DESIGN
http://www.e-welcom.com/
ISO14001Certified

Mirror Browser
for
Windows Mobile 6.5
Windows Embedded Handheld 6.5

CP30 Series
CP50 Series
CP60 Series
9200 Series

人にやさしく、地球にやさしく
Thoughtfully, Living with the Earth
WELCOM DESIGN
http://www.e-welcom.com/
ISO14001Certified

ウェルコムデザイン株式会社
本社 神戸市西区井吹台東町1-1-1 西神南センタービル 〒651-2242
Phone. 078-993-6010(代) Fax. 078-993-6020
東京 東京都文京区湯島3-14-9 湯島ビル 〒113-0034
Phone. 03-3836-9411(代) Fax. 03-3836-9412

改訂記録	
改訂番号	改訂日
Rev.1.0	2014/08/08（初版）
Rev.1.1	2014/09/05 MIRROR Browser Ver.2.1.3 に対応

INDEX

1. はじめに

WEB ｱﾌﾟﾘｹｰｼｮﾝによる情報収集ｼｽﾃﾑには、ﾀｰﾐﾅﾙが持つﾘｰﾀﾞなどのﾘｿｰｽを制御するための機能が必須であるというお客様からのご要望に応え、Internet Exploere Mobile を元にしたｺﾝﾊﾟｸﾄな機能的なﾌﾞﾗｳｻﾞ MIRROR Browser をﾘﾘｰｽしました。企業の情報収集ｼｽﾃﾑに必須のﾘｰﾀﾞ ﾓｼﾞｭｰﾙの制御が行るため、使いやすい業務ｼｽﾃﾑを容易に構築していただけます。

MIRROR Browser で WEB ｱﾌﾟﾘｹｰｼｮﾝを構築することにより、遠く離れた WEB ｻｰﾊﾞｰへｸﾗｲｱﾝﾄﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀからﾘｰﾀﾞの読み取り情報を簡単に送信することが可能になります。

本書では、MIRROR Browser の使用方法や HTML ｽｸﾘﾌﾟﾄ、ﾃﾞｰﾀ収集を行うﾀﾞｲﾅﾐｯｸ HTML ﾄﾞｷｭﾒﾝﾄ作成に必要な API について説明しています。

下記に MIRROR Browser の主な特徴を列挙します。

- ✓ Windows Mobile 6.5 及び Windows Embedded Mobile 6.5 搭載の Ciherlab ｼﾘｰｽﾞ ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀに対応
- ✓ ﾘﾓｰﾄ WEB ｻｰﾊﾞｰ上の WEB ﾍﾟｰｼﾞのﾌﾞﾗｳｼﾞﾝｸﾞ
- ✓ ﾘﾓｰﾄ WEB ｻｰﾊﾞｰへのﾘｱﾙﾀｲﾑなﾃﾞｰﾀ更新
- ✓ ﾕｰｻﾞｰﾌﾚﾝﾄﾞﾘｰでｼﾝﾌﾟﾙなﾕｰｻﾞｰｲﾝﾀｰﾌｪｲｽ
- ✓ 読取ﾃﾞｰﾀの自動入力
- ✓ ﾜﾝｸﾘｯｸによるﾌﾙｽｸﾘｰﾝ最大化表示
- ✓ 認証ﾕｰｻﾞｰ以外のｽｸﾘｰﾝﾛｯｸ機能の提供
- ✓ ﾌﾟﾛﾌｧｲﾙとしても設定のｲﾝﾎﾟｰﾄ/ｴｸｽﾎﾟｰﾄ
- ✓ ﾃﾞﾊﾞｲｽ制御用 JavaScript API の提供
- ✓ SSL ｺﾐﾆｹｰｼｮﾝﾌﾟﾛﾄｺﾙをｻﾎﾟｰﾄ

本章をお読みになり、MIRROR Browser の世界をお楽しみください。MIRROR Browser がお客様の業務改善にお役に立てることを願っております。この度は、MIRROR Browser 及び CipherLab ﾓﾊﾞｲﾙﾀｰﾐﾅﾙをお選び頂き誠にありがとうございました。

ﾗｲｾﾝｽについて

MIRROR Browser を正規で利用する場合は、ﾗｲｾﾝｽの購入が必要です。ﾗｲｾﾝｽを入力していない状態では、20 分間の試用が可能です。ﾗｲｾﾝｽの発行は、弊社又は販売店までご依頼ください。

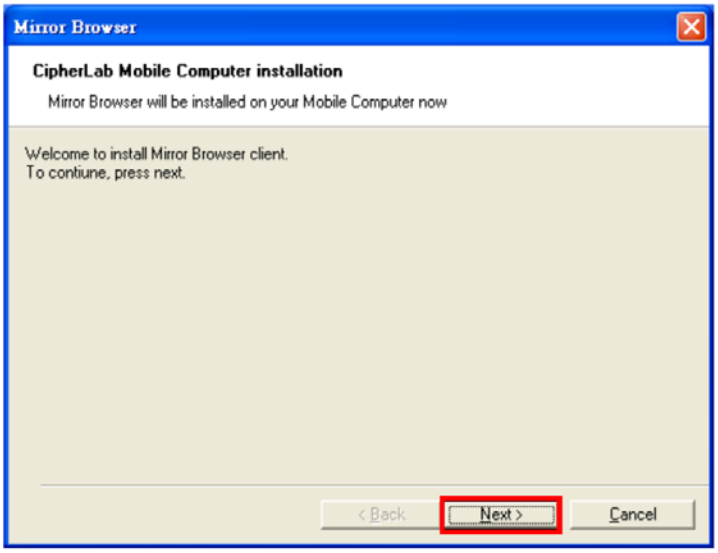
2. ｲﾝｽﾄｰﾙ

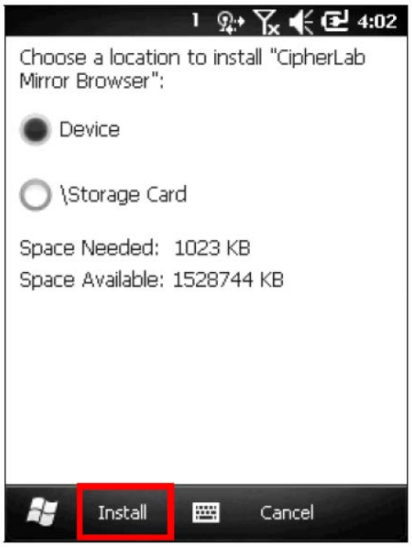
MIRROR Browser のｾｯﾄｱｯﾌﾟﾌｧｲﾙは、exe ﾌｧｲﾙと cab ﾌｧｲﾙの 2 種類が用意されています。各ﾌｧｲﾙを使って、Windows PC 及びﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀからｲﾝｽﾄｰﾙを実行できます。

2.1. PC でｲﾝｽﾄｰﾙ

下記にｲﾝｽﾄｰﾙ手順を示します。

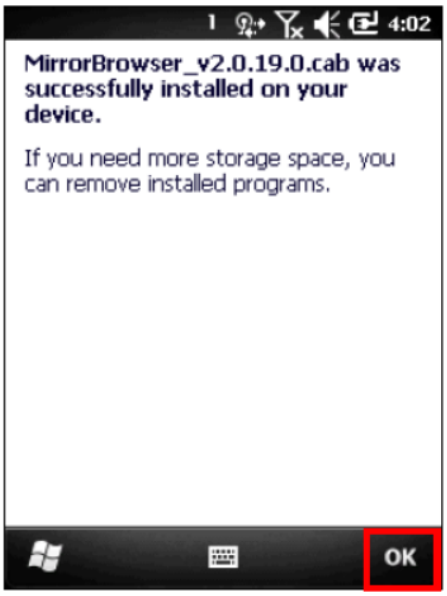
1. ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀと PC を USB 接続します。
2. PC で ActiveSync を実行し、ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀとﾊﾟｰﾄﾅｰｼｯﾌﾟを確立します。
3. PC でｾｯﾄｱｯﾌﾟﾌｧｲﾙ「MirrorBrowser_x.xxxx.exe」(X=ﾊﾞｰｼﾞｮﾝ)を実行します。





SD ｶｰﾄﾞが存在する場合は、ｲﾝｽﾄｰﾙ先の選択が行えます。ﾃﾞﾊﾞｲｽ内蔵ﾃﾞｨｽｸか、SD ｶｰﾄﾞかを選択して、次へ進んでください。

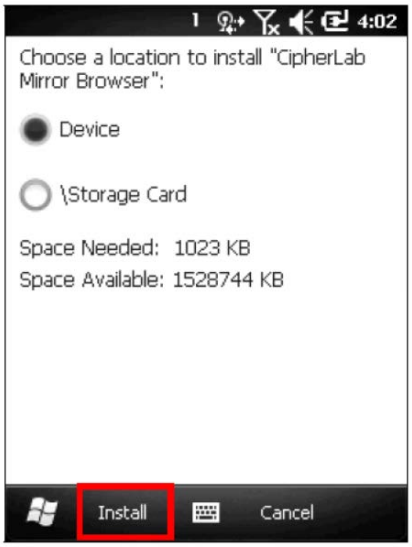
4. ｲﾝｽﾄｰﾙ完了のﾒｯｾｰｼﾞﾀﾞｲｱﾛｸﾞが表示されれば、「OK」ﾎﾞﾀﾝをﾀｯﾌﾟします。ｽﾀｰﾄｽｸﾘｰﾝに「MIRROR Browser」ｱｲｺﾝが表示されます。



2.2. ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀでｲﾝｽﾄｰﾙ

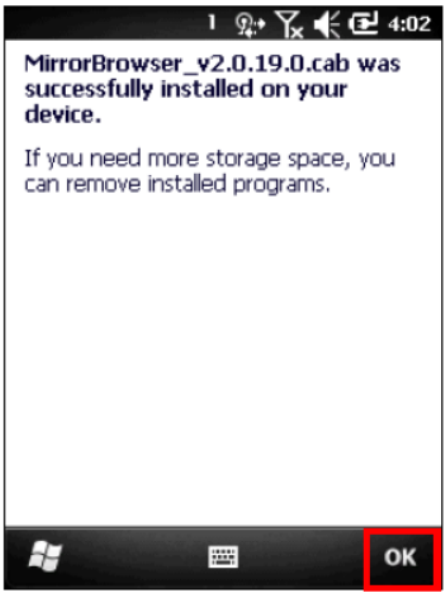
下記にｲﾝｽﾄｰﾙ手順を示します。

1. ｾｯﾄｱｯﾌﾟﾌｧｲﾙ「MirrorBrowser_x.xxxx.cab」(X=ﾊﾞｰｼﾞｮﾝ)ﾌｧｲﾙをﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀの内蔵ﾃﾞｨｽｸ又は SD ｶｰﾄﾞにｺﾋﾟｰします。
2. ﾌｧｲﾙｴｸｽﾌﾟﾛｰﾗで cab ﾌｧｲﾙを表示し、ﾀｯﾌﾟしてｾｯﾄｱｯﾌﾟを起動します。



SD ｶｰﾄﾞが存在する場合は、ｲﾝｽﾄｰﾙ先の選択が行えます。ﾃﾞﾊﾞｲｽ内蔵ﾃﾞｨｽｸか、SD ｶｰﾄﾞかを選択して、次へ進んでください。

3. ｲﾝｽﾄｰﾙ完了のﾒｯｾｰｼﾞﾀﾞｲｱﾛｸﾞが表示されれば、「OK」ﾎﾞﾀﾝをﾀｯﾌﾟします。ｽﾀｰﾄｽｸﾘｰﾝに「MIRROR Browser」ｱｲｺﾝが表示されます。



2.3. ｲﾝｽﾄｰﾙﾌｫﾙﾀﾞとﾌｧｲﾙ

MIRROR Browser は、ｲﾝｽﾄｰﾙﾌｫﾙﾀﾞの変更を行わない限り、ﾃﾞﾊﾞｲｽ内蔵ﾃﾞｨｽｸの「My Device¥Program¥Mirror Browser」にｲﾝｽﾄｰﾙされます。

下記にｲﾝｽﾄｰﾙされるﾌｧｲﾙの説明を行います。

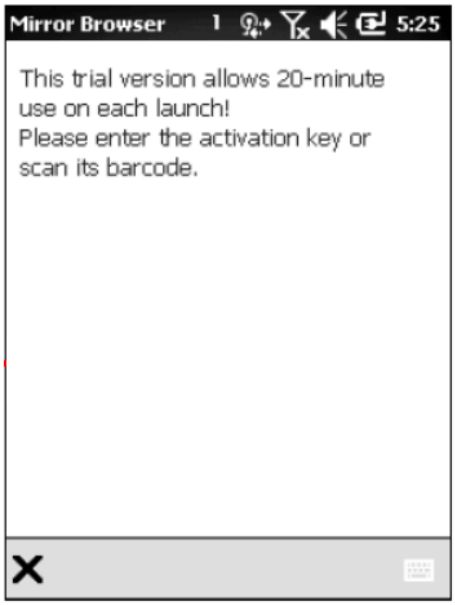
ﾌｧｲﾙ	説明
MirrorBrowser.exe	MIRROR Browser の実行ﾌｧｲﾙです。
MirrorBrowser.png	MIRROR Browser のｼｮｰﾄｶｯﾄに使われる画像ﾌｧｲﾙです。
test.htm	読み取りﾃｽﾄを行うためのﾃｽﾄ用 HTML ﾌｧｲﾙです。起動ﾍﾟｰｼﾞが設定されていない場合、この HTML ﾌｧｲﾙが自動的にﾛｰﾄﾞされます。
Browser_setting.ini	MIRROR Browser の設定内容が保存されます。ﾃﾞｨﾌｫﾙﾄ設定が変更された時点で、生成されるﾌｧｲﾙです。
Develop_reader.ini (CP30/50 のみ)	ﾘｰﾀﾞの設定内容が保存されます。ﾃﾞｨﾌｫﾙﾄ設定が変更された時点で、生成されるﾌｧｲﾙです。

2.4. MIRROR Browser の起動

下記に起動手順を示します。

1. ｽﾀｰﾄｽｸﾘｰﾝを表示します。
2. 「MIRROR Browser」ｱｲｺﾝをﾀｯﾌﾟします。

3. MIRROR Browser が起動し、test.htm が表示され、試用時間のｶｳﾝﾄが始まります。



ﾗｲｾﾝｽ登録されていない場合、MIRROR Browser を 20 分間、試用することができます。試用時間が経過すると、ﾗｲｾﾝｽ登録を求めるﾒｯｾｰｼﾞﾀﾞｲｱﾛｸﾞが表示されます。ﾗｲｾﾝｽの発行は、弊社又は販売店までご依頼ください。

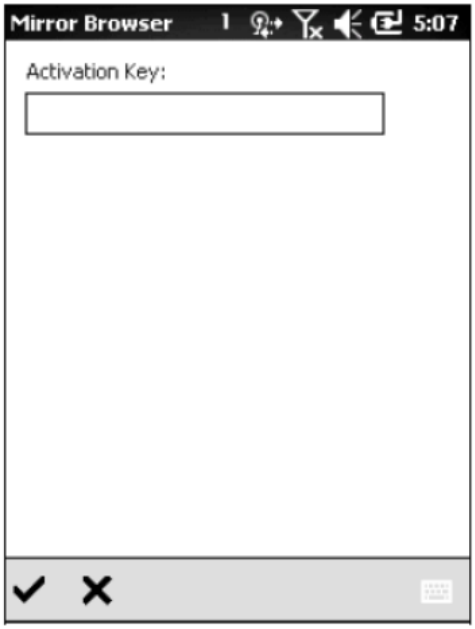
3. ﾗｲｾﾝｽ

3.1. ﾗｲｾﾝｽ登録

下記にﾗｲｾﾝｽ登録手順を示します。

1. MIRROR Browser を起動します。

2. ｿﾌﾄｷｰﾊﾞｰに表示されている「Menu」ﾎﾞﾀﾝをﾀｯﾌﾟし、ﾒﾆｭｰから「Activation」をﾀｯﾌﾟします。

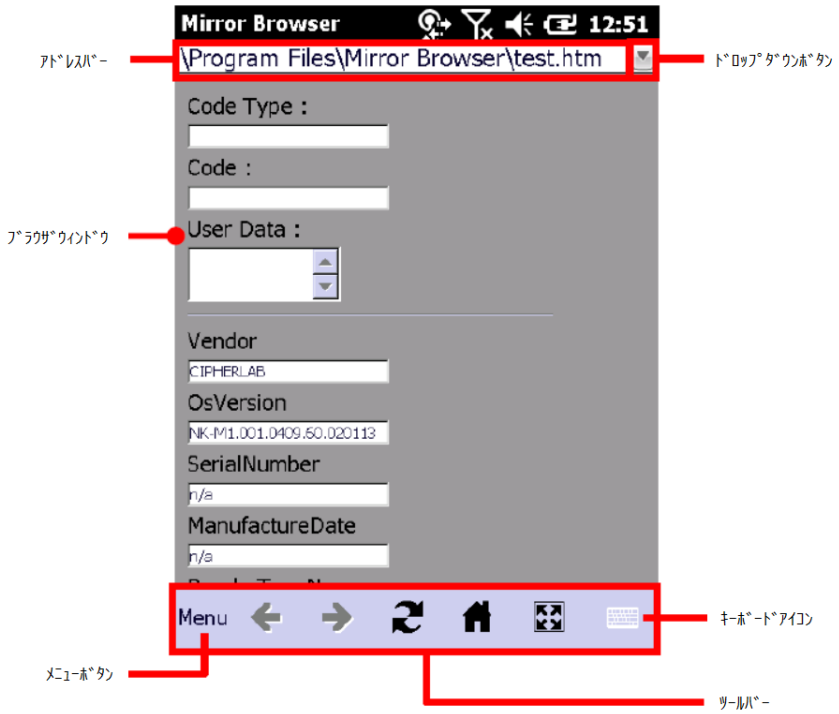


3. ﾗｲｾﾝｽ入力(Activation Key)画面が表示されるので、発行されたﾗｲｾﾝｽﾊﾞｰｺｰﾄﾞを読み取り「Enter」ｷｰを押します。(ｷｰﾎﾞｰﾄﾞからﾗｲｾﾝｽｷｰを手入力することも可能)
4. ﾗｲｾﾝｽ登録完了のﾒｯｾｰｼﾞﾀﾞｲｱﾛｸﾞが表示されてば「OK」ﾎﾞﾀﾝをﾀｯﾌﾟして完了です。制限なしに MIRROR Browser をご利用いただけます。

4. ﾕｰｻﾞｰｲﾝﾀｰﾌｪｲｽ

4.1. 画面の説明

下記は、ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀで、MIROR Browser を起動し、test.htm が読み込まれた画面です。



名称	説明
ﾌﾞﾗｳｻﾞｳｨﾝﾄﾞｳ	入力されたｱﾄﾞﾚｽ(URL)の内容を表示します。
ｱﾄﾞﾚｽﾊﾞｰ	表示したいｱﾄﾞﾚｽ(URL)を入力します。
ﾄﾞﾛｯﾌﾟﾀﾞｳﾝﾎﾞﾀﾝ	表示したｱﾄﾞﾚｽ履歴を表示します。
ﾒﾆｭｰﾎﾞﾀﾝ	MIRROR Browser のﾒﾆｭｰをｵｰﾌﾟﾝします。
ﾂｰﾙﾊﾞｰ	MIRROR Browser の機能にｸｲｯｸｱｸｾｽするためのｱｲｺﾝ群を表示します。

4.2. ﾒﾆｭｰﾎﾞﾀﾝ

ﾒﾆｭｰﾎﾞﾀﾝをﾀｯﾌﾟすると、MIRROR Browser で提供されているﾒﾆｭｰが表示されます。

ﾒﾆｭｰ	説明
Browser	ﾌﾞﾗｳｻﾞｳｨﾝﾄﾞｳを開きます。
Option	ﾌﾞﾗｳｻﾞｳｨﾝﾄﾞｳ、ﾘｰﾀﾞ、読み取りｺｰﾄﾞのｵﾌﾟｼｮﾝ設定を行います。
Activation	MIRROR Browser のﾗｲｾﾝｽ登録を行います。
About	MIRROR Browser の著作権やﾊﾞｰｼﾞｮﾝ情報を表示します。
Exit	MIRROR Browser を終了します。

4.3. ﾂｰﾙﾊﾞｰ

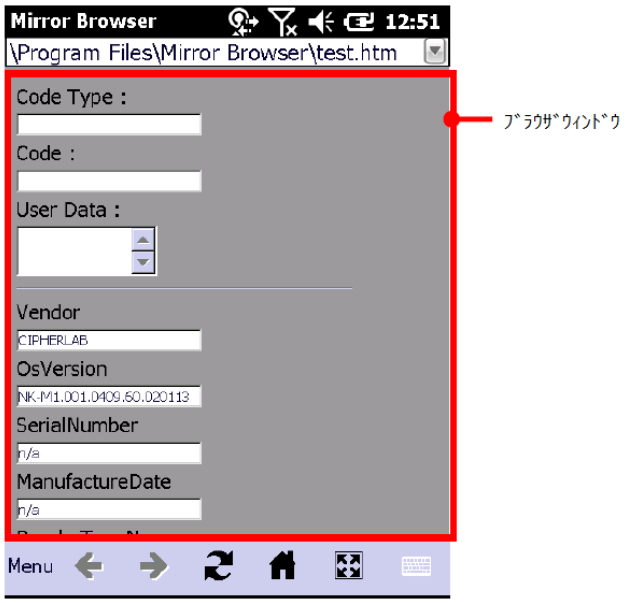
ﾂｰﾙﾊﾞｰは、MIRROR Browser が最大表示に切り替えられていない限り、画面の下部に表示されており、幾つかの機能に直接ｱｸｾｽすることができます。

ｱｲｺﾝ	説明
Menu	MIRROR Browser ﾒﾆｭｰを開きます。
	前のﾍﾟｰｼﾞを表示します。
	次のﾍﾟｰｼﾞを表示します。
	現在のﾍﾟｰｼﾞを再読み込みして表示します。
	ﾎｰﾑに設定されているﾍﾟｰｼﾞを開きます。
	MIRROR Browser を最大表示に切り替えます。ﾂｰﾙﾊﾞｰは非表示になります。
	ﾃｷｽﾄ入力を行うため、ｵﾝｽｸﾘｰﾝ・ｷｰﾎﾞｰﾄﾞを表示します。

4.4. ﾌﾞﾗｳｻﾞｳｨﾝﾄﾞｳ

指定されたｱﾄﾞﾚｽの内容を表示します。ﾎｰﾑｱﾄﾞﾚｽが設定されていない場合は、ﾃﾞｨﾌｫﾙﾄの「test.htm」が読み込まれ、表示されます。

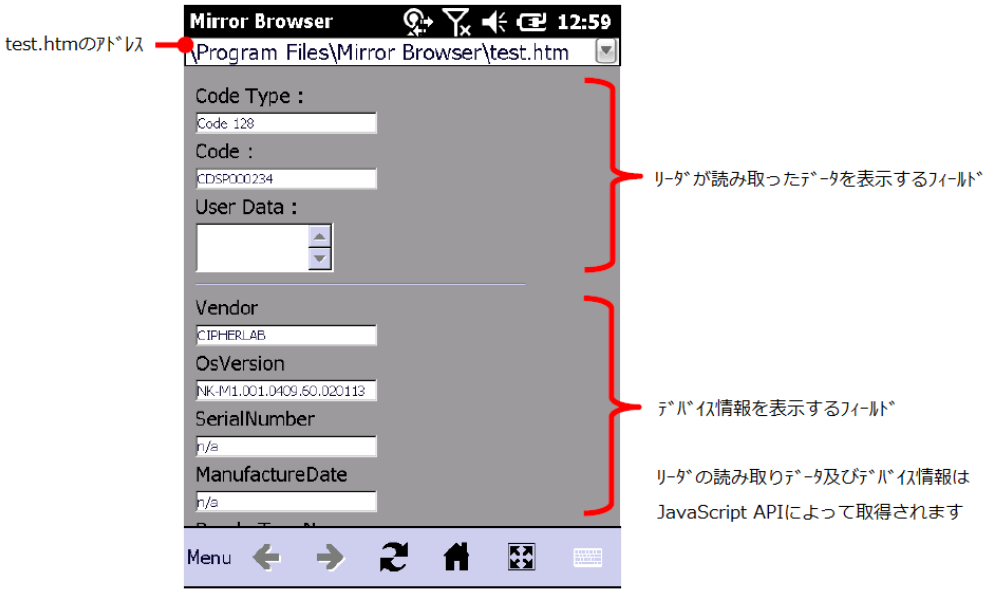
「test.htm」は、ｾｯﾄｱｯﾌﾟ時にｲﾝｽﾄｰﾙされるﾃｽﾄ用ﾌｧｲﾙです。



4.4.1. test.htm

「test.htm」は、そのﾌｧｲﾙ名の通り、ﾘｰﾀﾞがどう機能するかをﾃｽﾄするためのｼﾝﾌﾟﾙな入力ﾎﾞｯｸｽを装備したｻﾝﾌﾟﾙHTMLﾄﾞｷｭﾒﾝﾄﾌｧｲﾙです。

ﾘｰﾀﾞのﾃｽﾄ用途だけでなく、新規でﾃﾞｰﾀ収集ｱﾌﾟﾘｹｰｼｮﾝを作成する際のﾃﾝﾌﾟﾚｰﾄとしてもご利用ください。



ﾃﾞｰﾀの読み取りは、「SCAN」ｷｰ又はｻｲﾄﾞﾄﾘｶﾞｷｰで行います。

ﾌｨｰﾙﾄﾞ	説明
Code Type	読み取ったｺｰﾄﾞﾀｲﾌﾟを表示します。
Code	読み取ったﾃﾞｰﾀ(ﾊﾞｰｺｰﾄﾞ/RFID-UID)を表示します。
User Data	読み取ったRFIDﾕｰｻﾞｰﾃﾞｰﾀを表示します。

4.4.2. ﾌﾙｽｸﾘｰﾝﾓｰﾄﾞ

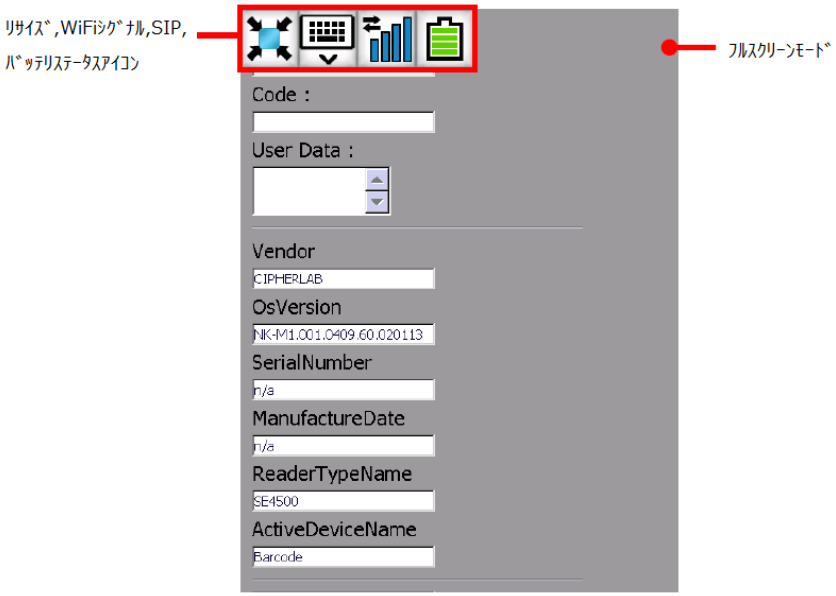
ﾌﾞﾗｳｻﾞｳｨﾝﾄﾞｳは、ﾀｽｸﾊﾞｰにある  ｱｲｺﾝをﾀｯﾌﾟすることで、最大化表示に切り替えることが可能です。

最大化表示に切り替わると、ﾃﾞｨﾌｫﾙﾄで、元のｻｲｽﾞに戻すための  ｱｲｺﾝが画面左上に表示されます。

ﾌﾙｽｸﾘｰﾝﾓｰﾄﾞでは、ﾂｰﾙﾊﾞｰ、ﾒﾆｭｰﾎﾞﾀﾝ及びｱﾄﾞﾚｽﾊﾞｰは全て表示されなくなります。画面最上部の OS ﾀｲﾄﾙﾊﾞｰについては、ﾕｰｻﾞｰが表示・非表示を設定することができます。また、代わりに、ﾊﾞｯﾃﾘｽﾃｰﾀｽｱｲｺﾝ及び WiFi ｼｸﾞﾅﾙｱｲｺﾝを表示する設定も可能です。

ﾊﾞｯﾃﾘｽﾃｰﾀｽｱｲｺﾝ 

WiFi ｼｸﾞﾅﾙｱｲｺﾝ 



ﾌﾙｽｸﾘｰﾝﾓｰﾄﾞから元に戻す

ﾘｻｲｽﾞ ｱｲｺﾝ  をﾀﾞﾌﾞﾙﾀｯﾌﾟすることで、元のｻｲｽﾞに戻すことができます。

ｵﾝｽｸﾘｰﾝ･ｷｰﾎﾞｰﾄﾞを表示する

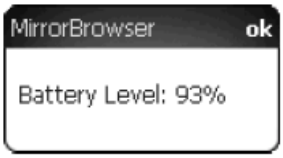
SIP ｱｲｺﾝ  をﾀﾞﾌﾞﾙﾀｯﾌﾟすることで、ｵﾝｽｸﾘｰﾝ･ｷｰﾎﾞｰﾄﾞを表示することができます。

ｱｲｺﾝ表示位置の変更

ﾃﾞｨﾌｫﾙﾄでは、ﾘｻｲｽﾞ･WiFi ｼｸﾞﾅﾙ･ﾊﾞｯﾃﾘｽﾃｰﾀｽｱｲｺﾝは、画面の左上に表示されます。これらのｱｲｺﾝは、ﾄﾞﾗｯｸﾞｱﾝﾄﾞﾄﾞﾛｯﾌﾟすることで、それぞれ別の位置に移動させることができます。

WiFi ｼｸﾞﾅﾙ&ﾊﾞｯﾃﾘｽﾃｰﾀｽ詳細表示

各ｱｲｺﾝをﾀﾞﾌﾞﾙﾀｯﾌﾟすることで、精度の高いﾊﾟｰｾﾝﾄ表示ﾀﾞｲｱﾛｸﾞをﾎﾟｯﾌﾟｱｯﾌﾟ表示することができます。

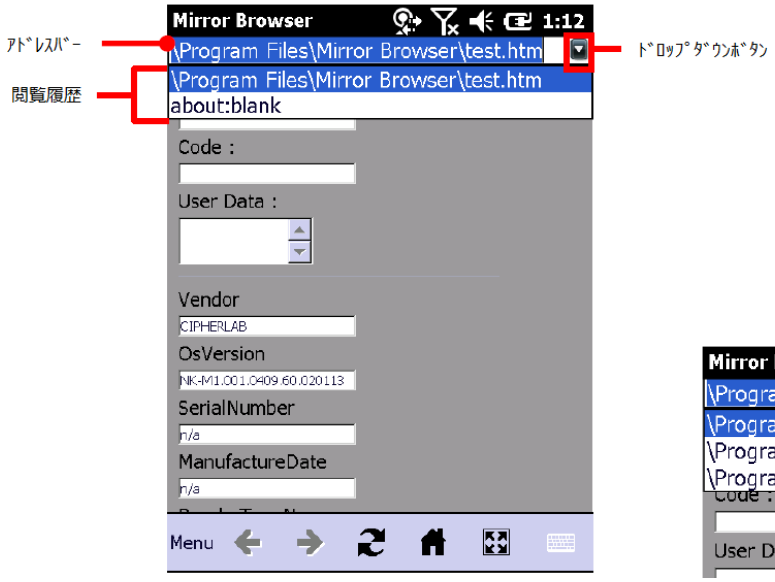


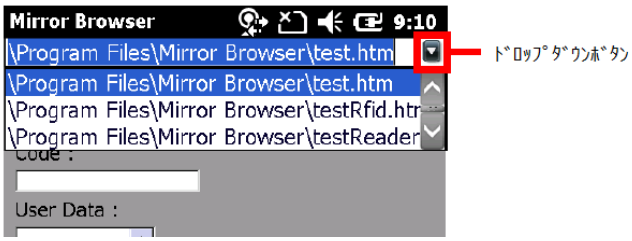
4.4.3. ｱﾄﾞﾚｽﾊﾞｰ&ﾄﾞﾛｯﾌﾟﾀﾞｳﾝﾎﾞﾀﾝ

ｱﾄﾞﾚｽﾊﾞｰは、WEB ﾍﾟｰｼﾞの URL ｱﾄﾞﾚｽを入力するためのﾃｷｽﾄﾎﾞｯｸｽです。ｱﾄﾞﾚｽﾊﾞｰは、ﾌﾞﾗｳｻﾞｳｨﾝﾄﾞｳを最大化表示に切り替えた場合、表示されません。

また、ｱﾄﾞﾚｽﾊﾞｰの右端に配置されたﾄﾞﾛｯﾌﾟﾀﾞｳﾝﾎﾞﾀﾝは、設定で「Lock the Home page at the next start」をﾁｪｯｸすることで無効となり、指定の URL ｱﾄﾞﾚｽ表示に固定することができます。

閲覧履歴として、ﾄﾞﾛｯﾌﾟﾀﾞｳﾝ表示されるのは、最新の 20 ｱﾄﾞﾚｽとなります。

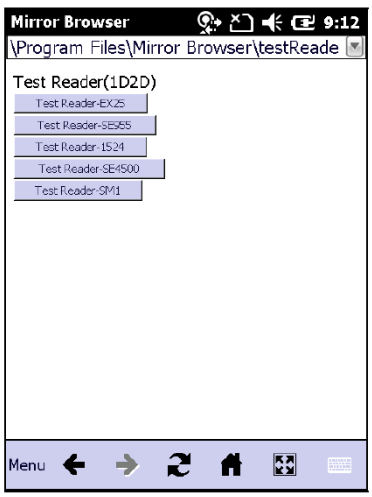


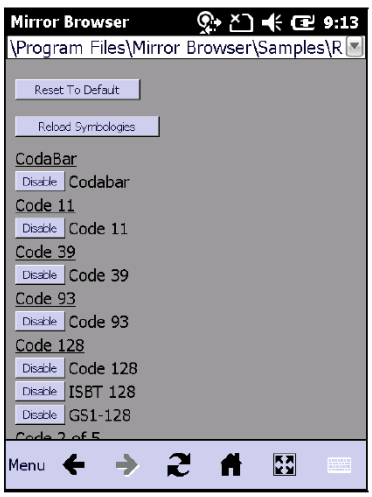


ｻﾝﾌﾟﾙ HTML ﾄﾞｷｭﾒﾝﾄ

3 種のｻﾝﾌﾟﾙ HTML ﾄﾞｷｭﾒﾝﾄが用意されています。これらは、ﾃﾞｨﾌｫﾙﾄで閲覧履歴に登録されているため、ﾄﾞﾛｯﾌﾟﾀﾞｳﾝﾎﾞﾀﾝで切り替えて表示することができます。

ﾌｨｰﾙﾄﾞ	説明
test.htm	ﾘｰﾀﾞの読み取りﾃﾞｰﾀやﾘｰﾀﾞﾀｲﾌﾟなどのﾃﾞﾊﾞｲｽ情報を表示するｻﾝﾌﾟﾙです。
testReader.htm	読み取りｺｰﾄﾞの設定を行うｻﾝﾌﾟﾙです。最初に実装されているﾘｰﾀﾞﾀｲﾌﾟの選択を行います。ﾘｰﾀﾞﾀｲﾌﾟは、test.htm で表示される正しいﾘｰﾀﾞﾀｲﾌﾟを選択してください。
testRFID.htm	特定の RFID ｶｰﾄﾞの読み取りや書き込みを行うｻﾝﾌﾟﾙです。RFID ﾘｰﾀﾞﾗｲﾀを搭載したﾓﾃﾞﾙで使用ください。





4.4.2. ﾎｯﾄｷｰ

下記のﾎｯﾄｷｰが使用可能です。

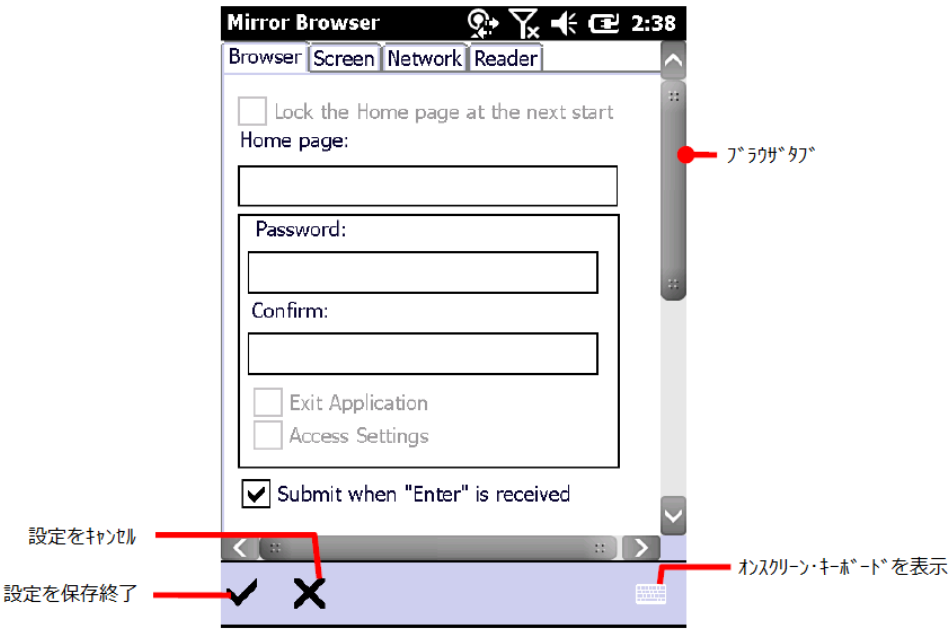
ﾎｯﾄｷｰ	説明
F1	ﾎｰﾑに設定されているﾍﾟｰｼﾞを開きます。  ｱｲｺﾝを同じ動作をします。
F5	ﾍﾟｰｼﾞの再読み込みを行います。  ｱｲｺﾝを同じ動作をします。
F10	設定ｳｨﾝﾄﾞｳを表示します。 「Menu」→「Options」→「Settings」と同じ動作をします。
F11	ｳｨﾝﾄﾞｳの最大表示と元のｻｲｽﾞ表示の切り替えを行います。  ｱｲｺﾝ、  ｱｲｺﾝと同じ動作をします。
Shift+UP	表示ﾌｫﾝﾄｻｲｽﾞを大きくします。
Shift+Down	表示ﾌｫﾝﾄｻｲｽﾞを小さくします。
Shift+Esc	ｵﾝｽｸﾘｰﾝ･ｷｰﾎﾞｰﾄﾞの表示・非表示を切り替えます。  ｱｲｺﾝを同じ動作をします。 このﾎｯﾄｷｰを有効にしたい場合は、「Menu」→「Options」→「Settings」の「Screen」ﾀﾌﾞで、「Set HW key…」にﾁｪｯｸを入れてください。

4.5. ｵﾌﾟｼｮﾝ→設定

ﾒﾆｭｰﾎﾞﾀﾝをﾀｯﾌﾟして、「Option」→「Setting」でｵﾌﾟｼｮﾝ設定ｳｨﾝﾄﾞｳを開きます。

4.5.1. Browser ﾀﾌﾞ

「Browser」ﾀﾌﾞでは、MIRROR Browser の動作ｵﾌﾟｼｮﾝを設定します。



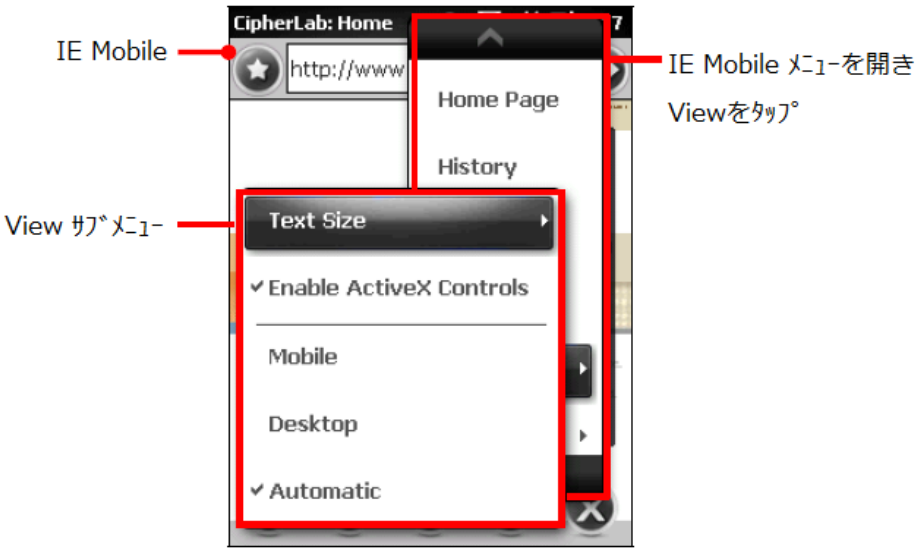
設定項目	説明
Lock the Home page at the next start	このｵﾌﾟｼｮﾝをﾁｪｯｸすると、ｱﾄﾞﾚｽﾊﾞｰが無効になり、ｱﾄﾞﾚｽの入力ができなくなります。このｵﾌﾟｼｮﾝは、次の「Home page」ｵﾌﾟｼｮﾝに URL ｱﾄﾞﾚｽが指定されている場合にのみ設定可能です。 業務ｱﾌﾟﾘｹｰｼｮﾝで、指定 URL ｱﾄﾞﾚｽ以外にｱｸｾｽさせたくない場合に有効です。
Home page	ﾌﾞﾗｳｻﾞ起動時に開く URL ｱﾄﾞﾚｽを指定します。
Password	認証ﾕｰｻﾞｰ以外が設定変更を行えないようﾊﾟｽﾜｰﾄﾞを指定します。
Confirm	「Password」で入力したﾊﾟｽﾜｰﾄﾞを確認のため、再入力します。
Exit application	ﾌﾞﾗｳｻﾞ終了時にﾊﾟｽﾜｰﾄﾞ入力を必要としたい場合は、ﾁｪｯｸします。
Access Settings	設定ｳｨﾝﾄﾞｳへのｱｸｾｽをﾊﾟｽﾜｰﾄﾞﾛｯｸしたい場合は、ﾁｪｯｸします。
Submit when “Enter” is received	このｵﾌﾟｼｮﾝをﾁｪｯｸすると、Enter ｷｰが押下された時点で、ﾃﾞｰﾀをｻｰﾊﾞｰに送信します。

ﾌﾞﾗｳｻﾞの一般設定

MIRROR Browser は、Internet Exploere Mobile(以下、IE Mobile)をﾍﾞｰｽとしているため、ﾌﾞﾗｳｻﾞの一般設定については、IE Mobile で行います。

下記の手順で、ﾌﾞﾗｳｻﾞの一般設定を行います。

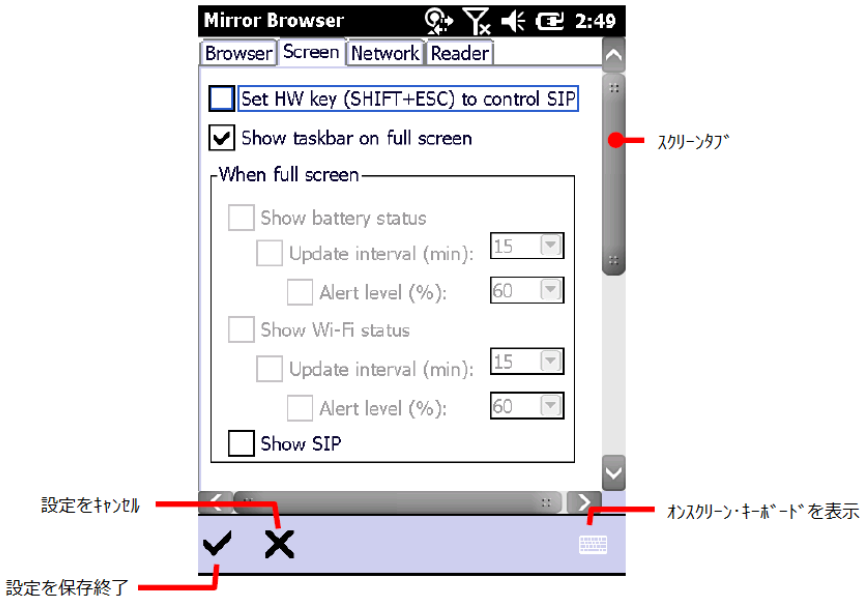
1. ｽﾀｰﾄｽｸﾘｰﾝから「IE Mobile」 ｱｲｺﾝをﾀｯﾌﾟします。
2. ﾂｰﾙｱｲｺﾝ を表示するため、画面右下の ｱｲｺﾝをﾀｯﾌﾟします。
3. ﾒﾆｭｰ ｱｲｺﾝをﾀｯﾌﾟし、「View」を選択し、操作に最適的な設定を行います。



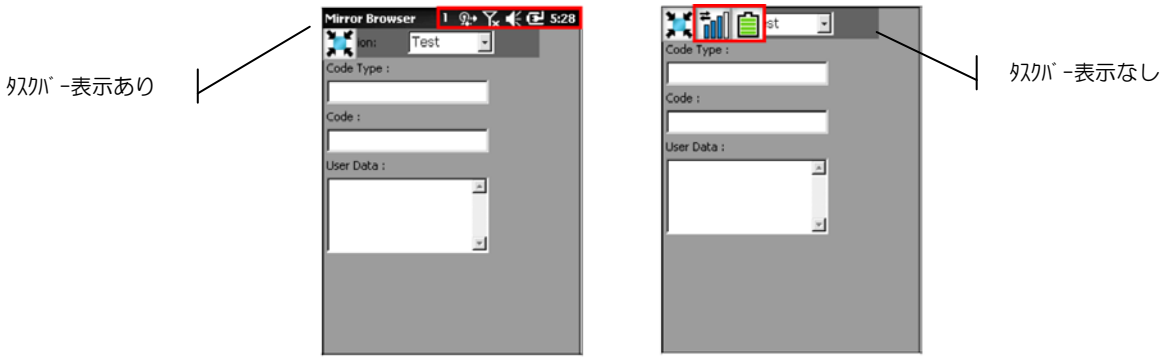
設定項目	説明
Text Size	ﾌﾞﾗｳｻﾞの表示ﾌｫﾝﾄｻｲｽﾞを設定します。
Enable ActiveX Controls	このｵﾌﾟｼｮﾝをﾁｪｯｸすることで、ｱﾆﾒｰｼｮﾝやﾎﾟｯﾌﾟｱｯﾌﾟﾒﾆｭｰなどよりｲﾝﾀｰﾗｸﾃｨﾌﾞで機能的にﾌﾞﾗｳｻﾞを活用することができます。
Mobile	ﾓﾊﾞｲﾙﾃﾞﾊﾞｲｽ用にﾃﾞｻﾞｲﾝされたWEBﾍﾟｰｼﾞを閲覧します。
Desktop	ﾃﾞｽｸﾄｯﾌﾟｺﾝﾋﾟｭｰﾀ用にﾃﾞｻﾞｲﾝされたWEBﾍﾟｰｼﾞを閲覧します。
Automatic	ﾓﾊﾞｲﾙﾃﾞﾊﾞｲｽ用とﾃﾞｽｸﾄｯﾌﾟｺﾝﾋﾟｭｰﾀ用にﾃﾞｻﾞｲﾝされたWEBﾍﾟｰｼﾞを自動的に切り替えて閲覧します。

4.5.2. Screen ﾀﾌﾞ

「Screen」ﾀﾌﾞ では、ﾌﾙｽｸﾘｰﾝﾓｰﾄﾞ のｽﾃｰﾀｽ表示方法を設定します。

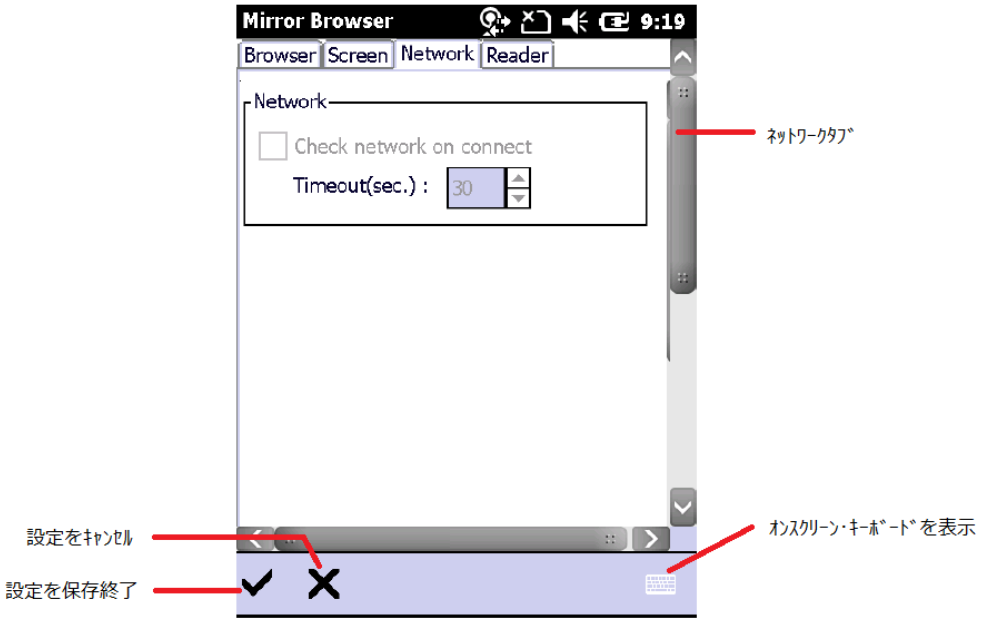


設定項目	説明
Set HW key(SHIFT+ESC) To control SIP	ﾎｯﾄｷｰ SHIFT+ESC を有効にするかを設定します。 有効にした場合、SHIFT+ESC でｵﾝｽｸﾘｰﾝ･ｷｰﾎﾞｰﾄﾞ の表示切り替えが行えます。
Show taskbar on full screen	ﾌﾙｽｸﾘｰﾝﾓｰﾄﾞ でﾀｽｸﾊﾞｰを表示するかを設定します。 ﾀｲﾄﾙﾊﾞｰには、WiFi ｼｸﾞﾅﾙ,電話ｽﾃｰﾀｽ,ﾎﾞﾘｭｰﾑｽﾃｰﾀｽ,ﾊﾞｯﾃﾘｽﾃｰﾀｽ,時刻情報などが表示されます。 このｵﾌﾟｼｮﾝをﾁｪｯｸした場合、次の「When full screen」ｵﾌﾟｼｮﾝは自動的に無効となります。
When full screen ｸﾞﾙｰﾌﾟﾎﾞｯｸｽ	このｸﾞﾙｰﾌﾟﾎﾞｯｸｽは、先のﾀｲﾄﾙﾊﾞｰがﾁｪｯｸされていない場合に表示されます。 Show battery status ﾊﾞｯﾃﾘｽﾃｰﾀｽを表示する場合は、ﾁｪｯｸします。 Update Interval(min) ﾊﾞｯﾃﾘｽﾃｰﾀｽを更新する間隔を 1,3,5,15,30,60 分の何れかに設定します。 Alert Level(%) ﾛｰﾊﾞｯﾃﾘ警告のﾎﾟｯﾌﾟｱｯﾌﾟﾀﾞｲｱﾛｸﾞを行う表示する基準ﾚﾍﾞﾙを 10,20,30,40,50,60,70,80,90%の何れかに設定します。 Show Wi-Fi status WiFi ｽﾃｰﾀｽを表示する場合は、ﾁｪｯｸします。 Update Interval(min) WiFi ﾃｰﾀｽを更新する間隔を 1,3,5,15,30,60 分の何れかに設定します。 Alert Level(%) WiFi ﾛｰｼｸﾞﾅﾙ警告のﾎﾟｯﾌﾟｱｯﾌﾟﾀﾞｲｱﾛｸﾞを行う表示する基準ﾚﾍﾞﾙを 10,20,30,40,50,60,70,80,90%の何れかに設定します。 Show SIP ｵﾝｽｸﾘｰﾝ･ｷｰﾎﾞｰﾄﾞ を表示する場合は、ﾁｪｯｸします。



4.5.3. Network ﾀﾌﾞ

「Network」ﾀﾌﾞでは、WiFi ﾈｯﾄﾜｰｸの接続確認を行うかを設定します。このｵﾌﾟｼｮﾝは、ﾗｲｾﾝｽ認証完了後、有効となります。

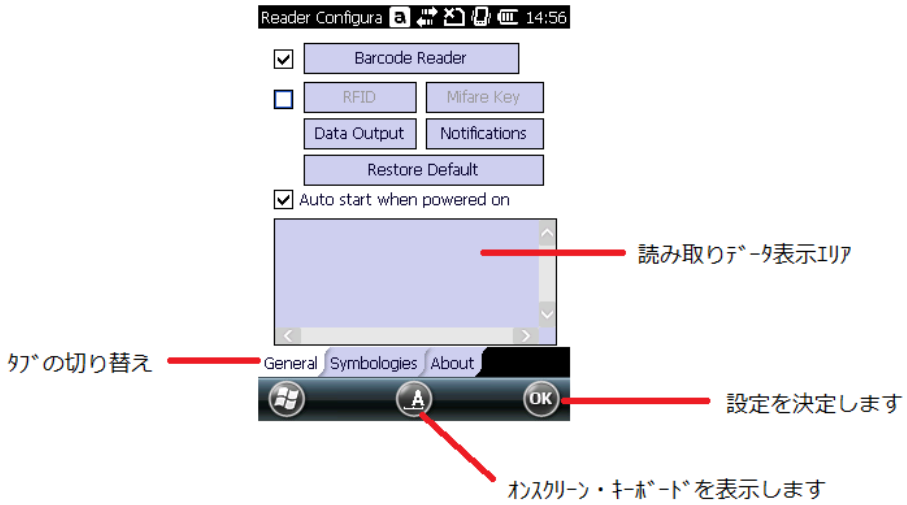


設定項目	説明
Network ｸﾞﾙｰﾌﾟﾎﾞｯｸｽ	Check network on connect WiFi ﾈｯﾄﾜｰの接続確認を行う場合は、ﾁｪｯｸします。 Timeout(sec.) ﾀｲﾑｱｳﾄ時間を秒単位で設定します。

4.5.4. Reader ﾀﾌﾞ

「Reader」ﾀﾌﾞ では、ﾘｰﾀﾞ の動作や読み取りｺｰﾄﾞ に関する様々な設定を設定します。「Reader」ﾀﾌﾞ をｸﾘｯｸすると、下記の画面表示に切り替わります。本書では、9200 ｼﾘｰｽﾞ ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀの画面を例として掲載しています。

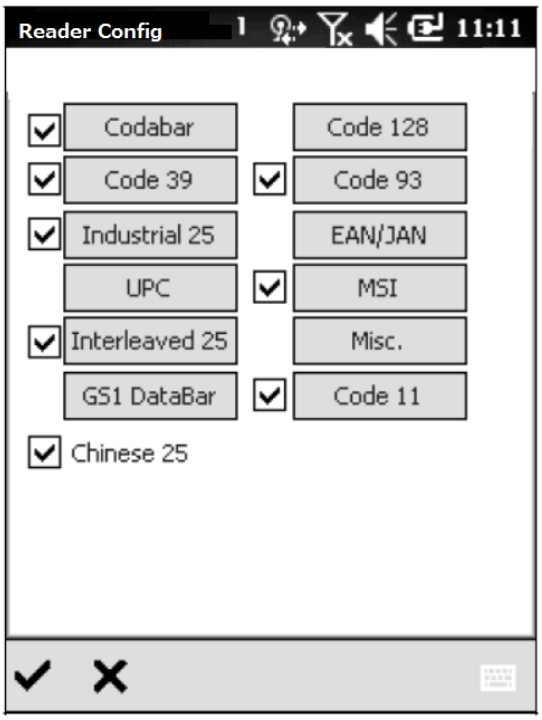
General ﾀﾌﾞ



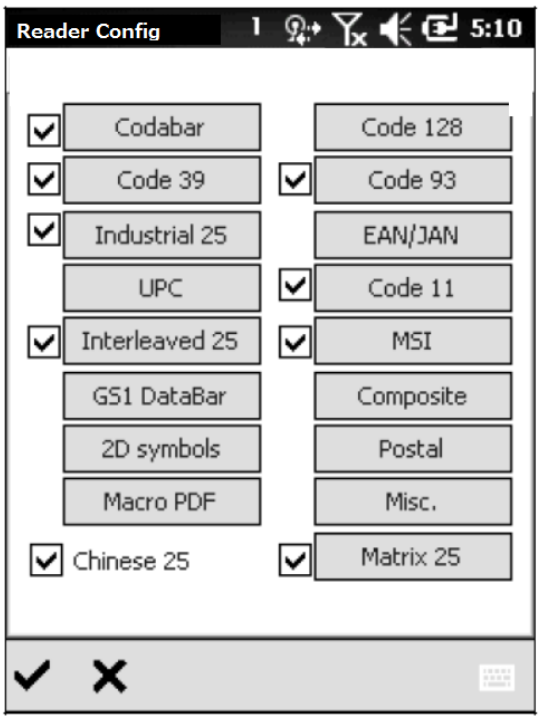
設定項目	説明
Barcode Reader	このｵﾌﾟｼｮﾝにﾁｪｯｸを入れると、ﾊﾞｰｺｰﾄﾞ ﾘｰﾀﾞ が使用可能になります。 「Barcode Reader」ﾎﾞﾀﾝをﾀｯﾌﾟすると、ﾊﾞｰｺｰﾄﾞ ﾘｰﾀﾞ の詳細設定画面に切り替ります。
RFID	このｵﾌﾟｼｮﾝにﾁｪｯｸを入れると、RFID ﾘｰﾀﾞ が使用可能になります。 「RFID」及び「Mifare Key」ﾎﾞﾀﾝをｸﾘｯｸすると、RFID 及び Mifare の詳細設定画面に切り替ります。
Data Output	ﾃﾞｰﾀ出力に関する詳細設定画面に切り替わります。
Notifications	ｻｳﾝﾄﾞ , ﾊﾞｲﾌﾞﾚｰｼｮﾝ,LED などｲﾝﾃﾞｨｹｰﾀの設定画面に切り替わります。
Restore Default	工場出荷時のﾃﾞｨﾌｫﾙﾄ設定値に戻します。
Auto start when powered on	このｵﾌﾟｼｮﾝにﾁｪｯｸを入れると、ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀの電源を立ち上げると自動的に「Reader Configuration」ﾌﾟﾛｸﾞﾗﾑがﾊﾞｯｸｸﾞﾗﾝﾄﾞで実行され、読み取りﾃﾞｰﾀをｷｰﾎﾞｰﾄﾞﾃﾞｰﾀとして、ｱｸﾃｨﾌﾞなｱﾌﾟﾘｹｰｼｮﾝに入力できます。
読み取りﾃﾞｰﾀ表示ｴﾘｱ	ﾘｰﾀﾞ でｺｰﾄﾞ を読み取ると、読み取りﾃﾞｰﾀが表示されます。設定内容が正しいかのﾁｪｯｸに利用します。

Symbologies ﾀﾌﾞ

1Dﾘｰﾀﾞﾓﾃﾞﾙ



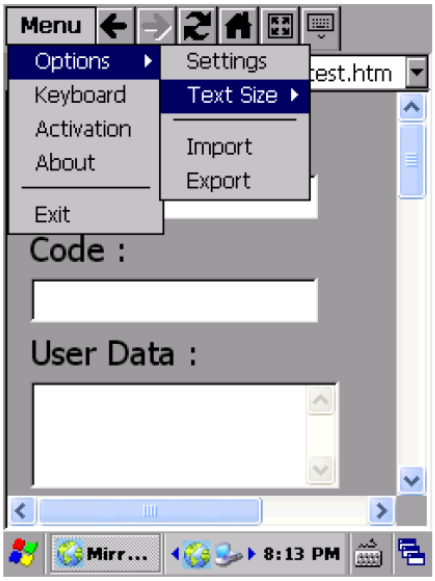
2Dﾘｰﾀﾞﾓﾃﾞﾙ

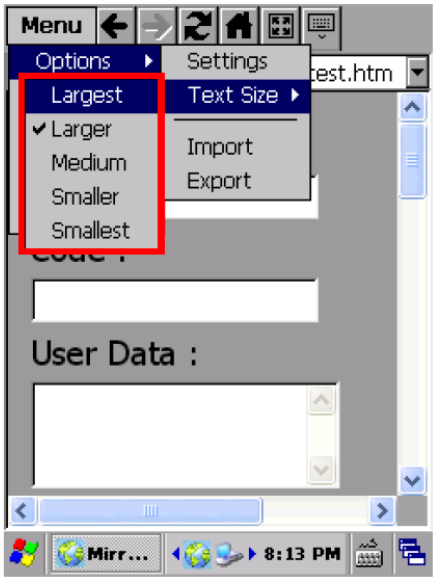


各ｼﾝﾎﾞﾙのﾎﾞﾀﾝをﾀｯﾌﾟすることで、詳細設定が行えます。

4.6. ｵﾌﾟｼｮﾝ→ﾃｷｽﾄｻｲｽﾞ

ﾒﾆｭｰﾎﾞﾀﾝをﾀｯﾌﾟして、「Option」→「Text Size」でﾌﾞﾗｳｻﾞの表示ﾌｫﾝﾄｻｲｽﾞを設定します。



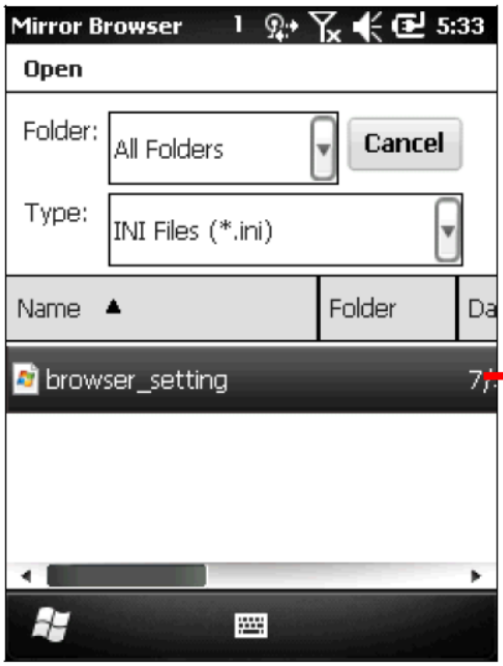


ﾌｫﾝﾄｻｲｽﾞ	説明
Largest	ﾌｫﾝﾄｻｲｽﾞ 最大
Larger	ﾌｫﾝﾄｻｲｽﾞ 大
Medium	ﾌｫﾝﾄｻｲｽﾞ 中
Smaller	ﾌｫﾝﾄｻｲｽﾞ 小
Smallest	ﾌｫﾝﾄｻｲｽﾞ 最小

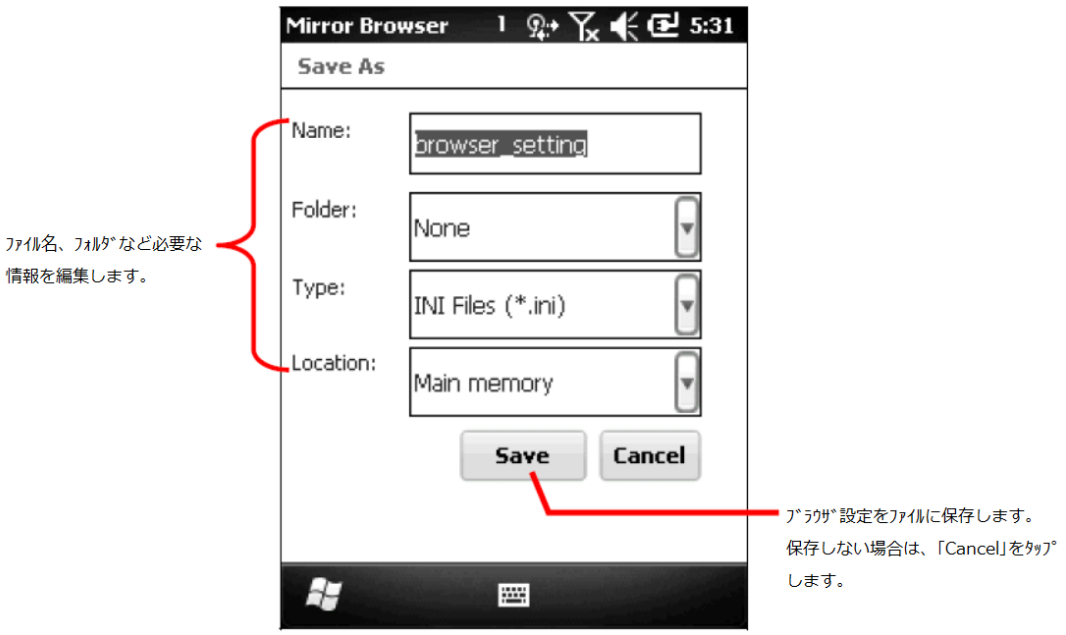
4.7. ｵﾌﾟｼｮﾝ→ｲﾝﾎﾟｰﾄ/ｴｸｽﾎﾟｰﾄ

ﾒﾆｭｰﾎﾞﾀﾝをﾀｯﾌﾟして、「Option」→「Import」及び「Option」→「Export」でﾌﾞﾗｳｻﾞ設定のｲﾝﾎﾟｰﾄ/ｴｸｽﾎﾟｰﾄが行えます。
ﾌﾞﾗｳｻﾞ設定は、.iniﾌｧｲﾙとして、保存されますが、機種により、ﾘｰﾀﾞ設定は対象外となります。

ｲﾝﾎﾟｰﾄ



ｴｸｽﾎﾟｰﾄ



ﾓﾃﾞﾙ名	説明
CP30/50	ﾘｰﾀﾞ設定も対象となります。
CP60/9200	ﾘｰﾀﾞ設定は対象外です。

5. HTMLﾄﾞｷｭﾒﾝﾄの開発

独自のHTMLﾄﾞｷｭﾒﾝﾄをｺｰﾃﾞｨﾝｸﾞすることにより、業務に合わせた最適なｱﾌﾟﾘｹｰｼｮﾝを開発することができます。
本章では、ﾌﾟﾛｸﾞﾗﾐﾝｸﾞに必要な基礎知識とﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀﾘｿｰｽを制御するための JavaScript API について説明します。

5.1. ﾃﾞｰﾀ収集ｽｸﾘﾌﾟﾄ&ﾌｫｰﾑ

ﾃﾞｰﾀ収集ｽｸﾘﾌﾟﾄとﾌｫｰﾑのｻﾝﾌﾟﾙは、「test.htm」として提供されています。

ﾊﾞｰｺｰﾄﾞ読み取りからﾃﾞｰﾀ取得までの基本的な流れは、下記の通りです。

1. ﾄﾘｶﾞｷｰを押して、目的のｺｰﾄﾞを読み取ります。
2. 読み取りに成功すると、onScanBarcode()関数が実行されます。
3. 読み取りに失敗した場合は、onScanBarcodeError()関数が実行されます。

test.htm

```
<html>
<meta name="viewport"
 content="width=device-width,initial-scale=1,maximum-scale=1,minimum-scale=1,user-scalable=no" />
<meta name="apple-mobile-web-app-capable" content="yes" />

<body onLoad="OnLoad();" bgcolor="#999999">


        <script>
                var m_bKeyboardEmulationEnabled=false;

           function OnLoad()
           {
                        this.init_value();
        }

                function init_value()
                {
                        getVendor();
                        getOsVersion();
                        getSerialNumber();
                        getManufactureDate();
                        getReaderTypeName();
                        getActiveDeviceName();

                        IsKeyboardEmulationEnabled();
                        GetPrefixCode();
                        GetSuffixCode();
                }

                function onScanBarcode(codeType, code)
                {
                        txtCodeType.value = codeType;
                        txtCode.value = code;
                        txtUserData.value = "";
                }

                function onScanRFID(codeType, code, userData)
                {
                        txtCodeType.value = codeType;
                        txtCode.value = code;
                txtUserData.value = userData;
        }

                function startVibration()
                {
                        var value = txt_startVibration.value;
                        window.external.startVibration(parseInt(value));
                }

                function playSound1()
                {
                        var value1 = txt_playSound1.value;
                        var value2 = "";

                        window.external.playSound(parseInt(value1),value2);
```

```
		}

		function playSound2()
		{
			var value1 = -1;
			var value2 = txt_playSound2.value;

			window.external.playSound(parseInt(value1),value2);
		}

		function EnableKeyboardEmulation()
		{
			//alert('EnableKeyboardEmulation()');
			var bEnable =
 !m_bKeyboardEmulationEnabled;//document.getElementById("chkEnableKeyboardEmulation").checked;
			//alert('EnableKeyboardEmulation.bEnable='+bEnable);
			window.external.EnableKeyboardEmulation(bEnable);
			IsKeyboardEmulationEnabled();
		}

		function IsKeyboardEmulationEnabled()
		{
			//alert('IsKeyboardEmulationEnabled()');
			var bEnabled = window.external.IsKeyboardEmulationEnabled();
			m_bKeyboardEmulationEnabled=bEnabled;
			//alert('IsKeyboardEmulationEnabled.bEnable='+bEnabled);
			if(bEnabled==true)
				document.getElementById("btnEnableKeyboardEmulation").value="Disable
KeyboardEmulation";
			else
				document.getElementById("btnEnableKeyboardEmulation").value="Enable
KeyboardEmulation";
			//document.getElementById("chkEnableKeyboardEmulation").checked=bEnabled;
		}

		function SetPrefixCode()
		{
			var code = txt_PrefixCode.value;
			window.external.SetPrefixCode(code);
		}

		function GetPrefixCode()
		{
			//alert('GetPrefixCode()');
			var code = window.external.GetPrefixCode();
			//alert('GetPrefixCode.code='+code);
			txt_PrefixCode.value = code;
			//alert('GetPrefixCode end');
		}

		function SetSuffixCode()
		{
			var code = txt_SuffixCode.value;
			window.external.SetSuffixCode(code);
		}

		function GetSuffixCode()
		{
			var code = window.external.GetSuffixCode();
			txt_SuffixCode.value = code;
		}


		function ToNA(value)
		{
			if(value=="")
				value="n/a";

			return value;
		}

		function getVendor()                                    {try{var
value=window.external.getVendor();                              txt_getVendor.value=ToNA(value);
		}catch(err){txt_getVendor.value=err;                                    }}
		function getOsVersion()                                         {try{var
value=window.external.getOsVersion();                           txt_getOsVersion.value=ToNA(value);
			}catch(err){txt_getOsVersion.value=err;
	}}
		function getSerialNumber()                              {try{var
value=window.external.getSerialNumber();                  txt_getSerialNumber.value=ToNA(value);
	}catch(err){txt_getSerialNumber.value=err;                              }}
```

```
                function getManufactureDate()                       {try{var
value=window.external.getManufactureDate();         txt_getManufactureDate.value=ToNA(value);
        }catch(err){txt_getManufactureDate.value=err;                       }}
                function getReaderTypeName()                        {try{var
value=window.external.getReaderTypeName();                  txt_getReaderTypeName.value=ToNA(value);
        }catch(err){txt_getReaderTypeName.value=err;                        }}
                function getActiveDeviceName()                      {try{var
value=window.external.getActiveDeviceName();                txt_getActiveDeviceName.value=ToNA(value);
        }catch(err){txt_getActiveDeviceName.value=err;                      }}
        </script>

        <table border="0" width="80%" bgcolor="#999999" align="left">
                <tr><td nowrap class="span_lab">Code Type :</td></tr>
                <tr><td     nowrap><input     type="text"     class="style_text"     name="txtCodeType"
maxlength="40"></td></tr>
                <tr><td nowrap class="span_lab">Code :</td></tr>
      <tr><td nowrap><input type="text" class="style_text" name="txtCode" maxlength="40"></td></tr>
                <tr><td nowrap class="span_lab">User Data :</td></tr>
      <tr><td        nowrap><textarea        class="style_text"        name="txtUserData"        cols="15"
rows="3"></textarea></td></tr>
                <tr><td><HR size="2"> </td></tr>
                <tr><td nowrap class="span_lab">Vendor</td></tr>
                <tr><td nowrap><input type="text" class="style_text" name="txt_getVendor" maxlength="40"
value="n/a"></td></tr>
                <tr><td nowrap class="span_lab">OsVersion</td></tr>
                <tr><td nowrap><input type="text" class="style_text" name="txt_getOsVersion" maxlength="30"
value="n/a"></td></tr>
                <tr><td nowrap class="span_lab">SerialNumber</td></tr>
                <tr><td   nowrap><input   type="text"   class="style_text"   name="txt_getSerialNumber"
maxlength="30" value="n/a"></td></tr>
                <tr><td nowrap class="span_lab">ManufactureDate</td></tr>
                <tr><td   nowrap><input   type="text"     class="style_text"name="txt_getManufactureDate"
maxlength="30" value="n/a"></td></tr>
                <tr><td nowrap class="span_lab">ReaderTypeName</td></tr>
                <tr><td   nowrap><input   type="text"   class="style_text"   name="txt_getReaderTypeName"
maxlength="30" value="n/a"></td></tr>
                <tr><td nowrap class="span_lab">ActiveDeviceName</td></tr>
                <tr><td   nowrap><input   type="text"   class="style_text"   name="txt_getActiveDeviceName"
maxlength="30" value="n/a"></td></tr>
                <tr><td><HR size="2"></td></tr>
                <tr><td   nowrap><input   type="text"   class="style_text"   name="txt_startVibration"
maxlength="1" value="1"></td></tr>
                <tr><td  nowrap><input  type="button"  class="style_button"  onClick="startVibration();"
value="startVibration"/></td></tr>
                <tr><td nowrap><input type="text" class="style_text" name="txt_playSound1" maxlength="1"
value="1"></td></tr>
                <tr><td   nowrap><input   type="button"   class="style_button"   onClick="playSound1();"
value="playSound"/></td></tr>
                <tr><td nowrap><input type="text" class="style_text" name="txt_playSound2" maxlength="30"
value="¥Windows¥Default"></td></tr>
                <tr><td   nowrap><input   type="button"   class="style_button"   onClick="playSound2();"
value="playSound"/></td></tr>

                <tr><td><HR size="2"></td></tr>
                <tr><td nowrap><input type="button" class="style_button" name="btnEnableKeyboardEmulation"
onclick="EnableKeyboardEmulation();"></td></tr>
                <tr><td nowrap><input type="text" name="txt_PrefixCode" maxlength="10" value=""></td></tr>
                <tr><td   nowrap><input   type="button"   class="style_button"   onClick="SetPrefixCode();"
value="SetPrefixCode"/></td></tr>
                <tr><td nowrap><input type="text" name="txt_SuffixCode" maxlength="10" value=""></td></tr>
                <tr><td   nowrap><input   type="button"   class="style_button"   onClick="SetSuffixCode();"
value="SetSuffixCode"/></td></tr>
        </table>

</body>
</html>
```

5.2. JavaScript API

ﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀのﾘｿｰｽを制御するための API が提供されています。

ｼｽﾃﾑ情報

getManufactureDate

用途	製造日を取得します。
書式	Result = window.external.getManufactureDate()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload="test()">
	<SCRIPT>
	function test() {
	y = window.external.getManufactureDate()
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、文字列で返されます。

getSerialNumber

用途	ｼﾘｱﾙ番号を取得します。
書式	Result = window.external.getSerialNumber()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload="test()">
	<SCRIPT>
	function test() {
	y = window.external.getSerialNumber()
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、文字列で返されます。

getVendor

用途	製造者を取得します。
書式	Result = window.external.getVendor()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload="test()">
	<SCRIPT>
	function test() {
	y = window.external.getVendor()
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、文字列(CIPHERLAB)で返されます。

getWiFiStatus

用途	WiFi ｽﾃｰﾀｽを取得します。
書式	Result = window.external.getWiFiStatus()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.getWiFiStatus()
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、0~100 の数値で返されます。数字が大きいほど、WiFi ｼｸﾞﾅﾙが強いことを意味します。

getPowerStatus

用途	電池残量を取得します。
書式	Result = window.external.getPowerStatus()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.getPowerStatus()
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、0~100 の数値で返されます。数字が大きいほど、電池残量が多いことを意味します。

読み取りｺｰﾄﾞ

enableSymbology

用途	読み取りｺｰﾄﾞを設定します。
書式	window.external.enableSymbology(name, enable)
引数	name　ｺｰﾄﾞ名を示す文字列を指定します。次頁のｺｰﾄﾞ名表を参照ください。 nable　読み取り有効/無効を True/False で指定します。 True　指定ｺｰﾄﾞの読み取りを有効に設定 False　指定ｺｰﾄﾞの読み取りを無効に設定

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	Var name = ‘Cadabar’;
	Var enable = True;
	window.external.enableSymbology(name, enable);
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

IsSymbologyEnabled

用途	指定ｺｰﾄﾞの読み取りｽﾃｰﾀｽを取得します。
書式	Result = window.external.IsSymbologyEnabled(name)
引数	name　ｺｰﾄﾞ名を示す文字列を指定します。次頁のｺｰﾄﾞ名表を参照ください。
ｺｰﾄﾞ例	<HTML> <BODY onload=”test()”> <SCRIPT> Var name = ‘Cadabar’; Var enable = window.external. IsSymbologyEnabled (name); ... } </SCRIPT> </BODY> </HTML>
備考	戻り値は、True/False で返されます。 True　指定ｺｰﾄﾞの読み取りは有効 False　指定ｺｰﾄﾞの読み取りは無効

ｺｰﾄﾞ名表						
ﾊﾞｰｺｰﾄﾞ		1Dﾘｰﾀﾞ			2Dﾘｰﾀﾞ	
ｺｰﾄﾞ名	API用ｺｰﾄﾞ名	SM1	SE955	SE965	SE4500	PL4507
ｺｰﾀﾞﾊﾞｰ(NW7	Codabar	○	○	○	○	○
ｺｰﾄﾞ11	Code11	×	○	○	○	○
ｺｰﾄﾞ39	Code39	○	○	○	○	○
ｺｰﾄﾞ93	Code93	○	○	○	○	○
ｺｰﾄﾞ128	Code128	○	○	○	○	○
	GS1128	○	○	○	○	○
	ISBT128	○	○	○	○	○
ｺｰﾄﾞ25	Chinese25	×	○	○	○	○
	Discrete25	○	○	○	○	○
	Interleaved25	○	○	○	○	○
	Matrix25	×	×	×	○	○
ｺﾝﾎﾟｼﾞｯﾄ	CompositeCCAB	×	×	×	○	○
	CompositeCCC	×	×	×	○	○
	CompositeTLC39	×	×	×	○	○
GS1 Databar	GS1DataBar14	○	○	○	○	○
	GS1DataBarLimited	○	○	○	○	○
	GS1DataBatExpanded	○	○	○	○	○
Korean35	Korean3of5	×	×	×	○	○
MSI	MSI	○	○	○	○	○
郵便ｺｰﾄﾞ	AustralianPostal	×	×	×	○	○
	JapanPostal	×	×	×	○	○
	NetherlandKIXCode	×	×	×	○	○
	USPostnet	×	×	×	○	○
	USPlanet	×	×	×	○	○
	UKPostal	×	×	×	○	○
UPC/EAN	Ean8	○	○	○	○	○
	Ean13	○	○	○	○	○
	UPCA	○	○	○	○	○
	UPCE	○	○	○	○	○
	UPCE1	○	○	○	○	○
二次元ｺｰﾄﾞ		1Dﾘｰﾀﾞ			2Dﾘｰﾀﾞ	
ｺｰﾄﾞ名	API用ｺｰﾄﾞ名	SM1	SE955	SE965	SE4500	PL4507
Aztec	Aztec	×	×	×	○	○
DataMatrix	DataMatrix	×	×	×	○	○
Maxicode	Maxicode	×	×	×	○	○
MicroPDF417	MicroPDF417	×	×	×	○	○
MicroQR	MicroQR	×	×	×	○	○
PDF417	PDF417	×	×	×	○	○
MicroQR	MicroQR	×	×	×	○	○
PDF417	PDF417	×	×	×	○	○
QRCode	QRCode	×	×	×	○	○

ﾘｰﾀﾞ

GetReaderType

用途	ﾘｰﾀﾞﾀｲﾌﾟを取得します。
書式	window.external.GetReaderType()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.GetReaderType();
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値については、ご利用のﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀのﾌﾟﾛｸﾞﾗﾐﾝｸﾞｶﾞｲﾄﾞを参照ください。

GetBcReaderType

用途	ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞﾀｲﾌﾟを取得します。
書式	window.external.GetBcReaderType()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.GetBcReaderType();
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値については、ご利用のﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀのﾌﾟﾛｸﾞﾗﾐﾝｸﾞｶﾞｲﾄﾞを参照ください。

GetRFIDReaderType

用途	RFIDﾘｰﾀﾞﾀｲﾌﾟを取得します。
書式	window.external.GetRFIDReaderType()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.GetRFIDReaderType();
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値については、ご利用のﾓﾊﾞｲﾙｺﾝﾋﾟｭｰﾀのﾌﾟﾛｸﾞﾗﾐﾝｸﾞｶﾞｲﾄﾞを参照ください。

GetReaderTypeName

用途	ﾘｰﾀﾞ ﾀｲﾌﾟ名を取得します。
書式	Result = window.external.GetRFIDReaderTypeName()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.GetReaderTypeName();
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、ﾘｰﾀﾞ ﾀｲﾌﾟ名を示す下記の文字列で返されます。 SE955E SE955I SE4500 SE965 SE4500+PL4507 Intermec EX25 SE1524 SM1 SE950 SE4407 SE4507 RFID

GetActiveDevice

用途	ｱｸﾃｨﾌﾞなﾘｰﾀﾞ ﾀｲﾌﾟを取得します。
書式	Result = window.GetActiveDevice()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.GetActiveDevice();
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、数値で返されます。 0 ｱｸﾃｨﾌﾞなﾘｰﾀﾞ無し 7 ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞ 32 RFIDﾘｰﾀﾞ 255 全ﾘｰﾀﾞ

GetActiveDeviceName

用途	ｱｸﾃｨﾌﾞなﾘｰﾀﾞ ﾀｲﾌﾟ名を取得します。
書式	Result = window.GetActiveDeviceName()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.GetActiveDeviceName();
	alert(y);
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、ﾘｰﾀﾞ ﾀｲﾌﾟ名を示す下記の文字列で返されます。 Barcode RFID Barcode,RFID none

enableBarcodeScanner

用途	ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞを有効にします。
書式	window.external.enableBarcodeScanner(n)
引数	下記の値を引数として指定します。 True　ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞを有効に設定 False　ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞを無効に設定

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	// ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞ有効
	window.external.enableBarcodeScanner(true);
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

IsBarcodeScannerEnabled

用途	ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞのｽﾃｰﾀｽを取得します。
書式	Result = window.external.IsBarcodeScannerEnabled()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.IsBarcodeScannerEnabled();
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、True/False で返されます。 True　ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞ有効 False　ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞ無効

enableRFIDScanner

用途	ﾊﾞｰｺｰﾄﾞﾘｰﾀﾞを有効にします。
書式	window.external.enableRFIDScanner(n)
引数	下記の値を引数として指定します。 True　RFIDﾘｰﾀﾞを有効に設定 False　RFIDﾘｰﾀﾞを無効に設定

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	// RFIDﾘｰﾀﾞ有効
	window.external.enableRFIDScanner(True);
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

IsRFIDScannerEnabled

用途	RFID ﾘｰﾀﾞのｽﾃｰﾀｽを取得します。
書式	Result = window.external.IsRFIDScannerEnabled()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.IsRFIDScannerEnabled();
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、True/False で返されます。 True RFID ﾘｰﾀﾞ有効 False RFID ﾘｰﾀﾞ無効

ReadFromDevice

用途	ﾃﾞﾊﾞｲｽからｼﾝﾎﾞﾛｼﾞｰ設定を読み出します。
書式	window.external.ReadFromDevice()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.ReadFromDevice();
	}
	...
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

WriteToDevice

用途	ｼﾝﾎﾞﾛｼﾞｰ設定をﾃﾞｲﾊﾞｽに書き込みます。
書式	window.external.WriteToDevice()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.WriteDevice();
	}
	...
	</SCRIPT>
	</BODY>
	</HTML>

備考	EnableAutoWriteToDevice が False に設定されている場合に、この関数をｺｰﾙします。

EnableAutoReadFromDevice

用途	ﾃﾞﾊﾞｲｽからｼﾝﾎﾞﾛｼﾞｰ設定を自動的に読み出すかを設定します。
書式	window.external.EnableAutoReadFromDevice(n)
引数	下記の値を引数として指定します。 True　自動読み出しを有効に設定 False　自動読み出しを無効に設定

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　window.external.EnableAutoReadFromDevice(True);
	　　...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

IsAutoReadFromDeviceEnabled

用途	AutoReadFromDevice ｽﾃｰﾀｽを取得します。
書式	Result = window.external.IsAutoReadFromDeviceEnabled()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　y = window.external.IsAutoReadFromDeviceEnabled();
	　　...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、True/False で返されます。 True　AutoReadFromDevice 有効 False　AutoReadFromDevice 無効

EnableAutoWriteToDevice

用途	ﾃﾞﾊﾞｲｽにｼﾝﾎﾞﾛｼﾞｰ設定を自動的に書き込むかを設定します。
書式	window.external.EnableAutoWriteToDevice(n)
引数	下記の値を引数として指定します。 True　自動書き込みを有効に設定 False　自動書き込みを無効に設定

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　window.external.EnableAutoWriteToDevice(True);
	　　...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

IsAutoWriteToDeviceEnabled

用途	AutoWriteToDevice ｽﾃｰﾀｽを取得します。
書式	Result = window.external.IsAutoWriteToDeviceEnabled()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　y = window.external. IsAutoWriteToDeviceEnabled();
	　　...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、True/False で返されます。 True　AutoWriteToDevice 有効 False　AutoWriteToDevice 無効

onScanBarcode

用途	ﾊﾞｰｺｰﾄﾞを読み取ります。
書式	function onScanBarcode(codeType, code)
引数	codeType　ｺｰﾄﾞﾀｲﾌﾟを示す 16 進数 2 文字がｾｯﾄされます。 Code　読み取りﾃﾞｰﾀがｾｯﾄされます。 codeType　ｺｰﾄﾞの種類 ‘40’　ISBT 128 ‘41’　ｺｰﾄﾞ 39 ‘42’　ｺｰﾄﾞ 32 ‘43’　CIP39 ‘44’　ｲﾝﾀﾞｽﾄﾘｱﾙ 25 ‘45’　ｲﾝﾀｰﾘｰﾌﾞﾄﾞ 25 ‘46’　ﾏﾄﾘｸｽ 25 ‘47’　ｺｰﾀﾞﾊﾞｰ(NW7) ‘48’　ｺｰﾄﾞ 93 ‘49’　ｺｰﾄﾞ 128 ‘4A’　UPC-E0 ‘4B’　UPC-E0 ｱﾄﾞｵﾝ 2 ‘4C’　UPC-E0 ｱﾄﾞｵﾝ 5 ‘4D’　EAN-8 ‘4E’　EAN-8 ｱﾄﾞｵﾝ 2 ‘4F’　EAN-8 ｱﾄﾞｵﾝ 5 ‘50’　EAN-13(UPC-A, CCD/ﾚｰｻﾞｴﾝｼﾞﾝの場合) ‘51’　EAN-13 ｱﾄﾞｵﾝ 2 ‘52’　EAN-13 ｱﾄﾞｵﾝ 5 ‘53’　MSI ‘54’　Plessey ‘55’　GS1-128 ‘56’　未定義 ‘57’　未定義 ‘58’　未定義 ‘59’　未定義 ‘5A’　Telepen ‘5B’　GS1 Databar Omnidirectional ‘5C’　GS1 Databar Limited ‘5D’　GS1 Databar Expanded ‘5E’　UPC-A ‘5F’　UPC-A ｱﾄﾞｵﾝ 2 ‘60’　UPC-A ｱﾄﾞｵﾝ 5 ‘61’　UPC-E1 ‘62’　UPC-E1 ｱﾄﾞｵﾝ 2 ‘63’　UPC-E1 ｱﾄﾞｵﾝ 5 ‘64’　TLC39 ‘65’　Trioptic ‘66’　Bookland ‘67’　ｺｰﾄﾞ 11 ‘68’　ｺｰﾄﾞ 39 ﾌﾙｱｽｷｰ ‘69'　IATA(ｺｰﾄﾞ 25) ‘6A’　ｲﾝﾀﾞｽﾄﾘｱﾙ 25 ‘6B’　PDF417 ‘6C’　microPDF417 ‘6D’　Data Matrix ‘6E’　Maxicode ‘6F’　QR

引数	‘70’ US Postnet ‘71’ US Planet ‘72’ UK Pastal ‘73’ 日本郵便ｺｰﾄﾞ(ｶｽﾀﾏﾊﾞｰｺｰﾄﾞ) ‘74’ Australian Postal ‘75’ Dutch Postal ‘76’ Composite Code ‘77’ Macro PDF ‘78’ Coupon Code ‘79’ Chinese 25 ‘7A’ Aztec ‘7B’ microQR ‘7C’ USPS 4CB/One Code/Intelligent Mail ‘7D’ UPU FICS Postal ‘7E’ Macro MicroPDF417

ｺｰﾄﾞ例	<HTML>
	<SCRIPT>
	function onScanBarcode(codeType, code) {
	if(codeType==’41’) {
	t1.value = “Code 39”
	t2.value = code;
	t4.value = “”;
	} else if(codeType==’47’) {
	t1.value = “Codabar(NW7)”
	t2.value = code;
	t4.value = “”;
	}
	}
	</SCRIPT>
	</HTML>

備考	ﾊﾞｰｺｰﾄﾞが読み取られると、HTMLﾄﾞｷｭﾒﾝﾄはこの関数を実行します。

onScanRFID

用途	RFID ﾀｸﾞ を読み取ります。
書式	function onScanRFID(codeType, code, userData)
引数	codeType　　ﾀｸﾞ ﾀｲﾌﾟ を示すｷｬﾗｸﾀ 1 文字がｾｯﾄされます。 Code　　読み取った UID がｾｯﾄされます。 userData　　読み取ったﾕｰｻﾞｰﾃﾞｰﾀがｾｯﾄされます。 codeType　　ﾀｸﾞ の種類 ‘T’　　Icode ‘M’　　Mifare Ultralight ISO 14443A ‘S’　　SR176 ‘T’　　Tagit ‘V’　　ISO 15693 ‘Z’　　ISO 14443B

ｺｰﾄﾞ 例	<HTML>
	<SCRIPT>
	function onScanRFID(codeType, code, userData) {
	　switch(codeType) {
	　　case ‘I’:
	　　t1.value = ‘Icode’;
	　　t2.value = code;
	　　t3.value = userData;
	　　break;
	　　default:
	　　t1.value = codeType;
	　　t2.value = code;
	　　t3.value = userData;
	　　break;
	　}
	}
	</SCRIPT>
	</HTML>

備考	RFID ﾀｸﾞ が読み取られると、HTML ﾄﾞｷｭﾒﾝﾄはこの関数を実行します。

playSound

用途	WAV ﾌｧｲﾙを再生します。
書式	window.external.playSound(soundidx, path)
引数	soundidx　　ｲﾝﾃﾞｸｽ番号(ﾌｧｲﾙ番号)を指定します。 path　　ﾌｧｲﾙﾊﾟｽを指定します。(soundidx=-1 の場合) soundidx　　説明 0　　ﾐｭｰﾄ 1 ~ 9　　WAV ﾌｧｲﾙ 1~9 -1　　ﾕｰｻﾞｰ定義の WAV ﾌｧｲﾙ(ﾌｧｲﾙﾊﾟｽ指定要)

ｺｰﾄﾞ 例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　window.external.playSound(1, ‘’)
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	ｲﾝﾃﾞｯｸｽ番号 1~9 で指定可能な WAV ﾌｧｲﾙが準備されています。それ以外の任意 WAV ﾌｧｲﾙを再生する場合は、ｲﾝﾃﾞｯｸｽ番号を-1 とし、第 2 引数でﾌｧｲﾙﾊﾟｽを指定します。

startVibration

用途	ﾊﾞｲﾌﾞﾚｰｼｮﾝを制御します。
書式	window.external.startVibration(enable)
引数	enable　　ﾊﾞｲﾌﾞﾚｰｼｮﾝを起動する時間を秒単位で指定します。 Enable　　説明 0　　ﾊﾞｲﾌﾞﾚｰｼｮﾝｵﾌ >=1　　指定秒数、ﾊﾞｲﾌﾞﾚｰｼｮﾝｵﾝ

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　// 1秒間 ﾊﾞｲﾌﾞﾚｰｼｮﾝｵﾝ
	　　window.external.startVibration(1)
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

EnableKeyboardEmulation

用途	ｷｰﾎﾞｰﾄﾞｴﾐｭﾚｰｼｮﾝを設定します。
書式	window.external.EnablKeyboardEmulation(n)
引数	下記の値を引数として指定します。 True　ｷｰﾎﾞｰﾄﾞｴﾐｭﾚｰｼｮﾝを有効に設定 False　ｷｰﾎﾞｰﾄﾞｴﾐｭﾚｰｼｮﾝを無効に設定

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　window.external.EnableKeyboardEmulation(True);
	　　...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	JavaScript API を使って読み取りﾃﾞｰﾀを取得する場合は、False に設定します。

IsKeyboardEmulationEnabled

用途	ｷｰﾎﾞｰﾄﾞｴﾐｭﾚｰｼｮﾝｽﾃｰﾀｽを取得します。
書式	Result = window.external.IsKeyboardEmulationEnabled()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.IsKeyboardEmulationEnabled();
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値は、True/False で返されます。 True　ｷｰﾎﾞｰﾄﾞｴﾐｭﾚｰｼｮﾝ有効 False　ｷｰﾎﾞｰﾄﾞｴﾐｭﾚｰｼｮﾝ無効

SetPrifixCode

用途	ﾌﾟﾘﾌｨｯｸｽを設定します。
書式	window.external.SetPrifixCode(sCode)
引数	sCode　ﾌﾟﾘﾌｨｯｸｽに設定したい文字列を指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.SetPrifixCode(ABC)
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

GetPrifixCode

用途	ﾌﾟﾘﾌｨｯｸｽを取得します。
書式	Result = window.external.GetPrifixCode()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	y = window.external.GetPrifixCode()
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値として、設定されているﾌﾟﾘﾌｨｯｸｽｺｰﾄﾞが文字列で返されます。

SetSuffixCode

用途	ﾌﾟﾘﾌｨｯｸｽを設定します。
書式	window.external.SetSuffixCode(sCode)
引数	sCode　　ｻﾌｨｯｸｽに設定したい文字列を指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　window.external.SetSuffixCode(XYZ)
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	無し

GetSuffixCode

用途	ﾌﾟﾘﾌｨｯｸｽを取得します。
書式	Result = window.external.GetSuffixCode()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　y = window.external.GetSuffixCode()
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	戻り値として、設定されているｻﾌｨｯｸｽｺｰﾄﾞが文字列で返されます。

RFID ﾘｰﾀﾞ

ReadRfidData

<table>
<tr><td>用途</td><td>GUI ﾎﾞﾀﾝをﾀｯﾌﾟして、RFID ﾀｸﾞからﾃﾞｰﾀを読み取ります。
WM_DECODEDATA に干渉されることはありません。</td></tr>
<tr><td>書式</td><td>window.external.ReadRfidData(nType, sStartBlock, nReadLen, nTimeout, sLoginKey, nLoginKeyType)</td></tr>
<tr><td>引数</td><td>
nType　読み取りを行うﾃﾞｰﾀを指定します。

1　READ_UID
2　READ_DATA

nStartBlock　読み取り開始位置を指定します。

-1　RFID ﾀｸﾞのﾃﾞｨﾌｫﾙﾄﾌﾞﾛｯｸから読み取りを開始します。

<table>
<tr><th>ﾀｸﾞﾀｲﾌﾟ</th><th>ｽﾀﾝﾀﾞｰﾄﾞ</th><th>ﾃﾞｨﾌｫﾙﾄ開始ﾌﾞﾛｯｸ</th></tr>
<tr><td>Mifare</td><td>ISO 14443A</td><td>4</td></tr>
<tr><td>SR176</td><td>ISO 14443B</td><td>4</td></tr>
<tr><td>ICODE SLI</td><td>ISO 15693</td><td>3</td></tr>
<tr><td>LRI512</td><td>ISO 15693</td><td>0</td></tr>
<tr><td>SRF55VxxP</td><td>ISO 15693</td><td>3</td></tr>
<tr><td>EM4135</td><td>ISO 15693</td><td>0</td></tr>
<tr><td>Tag-It</td><td>ISO 15693</td><td>0</td></tr>
<tr><td>その他</td><td>ISO 15693</td><td>0</td></tr>
<tr><td>ICODE</td><td>Phillips</td><td>5</td></tr>
</table>
nReadLen　読み取りを行うﾊﾞｲﾄ数を指定します。RFID ﾀｸﾞの種類により、指定できる最大値は異なります。最大値を超える値が指定された場合は、自動的に最大値に調整されます。

nTimeout　予備(現在は使用していません)

sLoginKey　Mifare Standard 1K/4K や SLE66R35 など特定の RFID ﾀｸﾞへｱｸｾｽするためのﾛｸﾞｲﾝｷｰを 12 ﾊﾞｲﾄ長の 16 進数表記で指定します。

<table>
<tr><th>ﾛｸﾞｲﾝｷｰ例</th><th>16 進数表記文字列</th></tr>
<tr><td>0</td><td>FFFFFFFFFFFF</td></tr>
<tr><td>F1F2F3F4F5F6</td><td>F1F2F3F4F5F6</td></tr>
</table>
nLoginKeyType　ﾛｸﾞｲﾝｷｰﾀｲﾌﾟを指定します。

<table>
<tr><th>値</th><th>ﾛｸﾞｲﾝｷｰﾀｲﾌﾟ</th></tr>
<tr><td>0</td><td>Key A</td></tr>
<tr><td>1</td><td>Key B</td></tr>
</table>
</td></tr>
</table>

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	Var nType = 1; // Read UID
	Var nStartBlock = -1;
	Var nReadLen = 10;
	Var nTimeout = 3;
	Var sLoginKey = ‘FFFFFFFFFFFF’;
	Var nLoginKeyType = 0;
	
	Var nReadRfidData = window.external.ReadRfidData(nType, nStartBlock, nReadLen, nTimeout, sLoginKey, nLoginKeyType);
	
	Alert(“ReadRfidUID=”+nReadRfidData);
	If(nType==1) {
	Var sCardUid = window.external.GetCardUid();
	}
	Else {
	Var sCardData = window.external.GetCardData();
	}
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、取得したﾊﾞｲﾄ数(最後の NULL を除く)を返します。 ｴﾗｰの場合は、下記のｴﾗｰｺｰﾄﾞ(ﾏｲﾅｽ値)を返します。 -101 E_READER_NOT_INIT -272 E_TAG_READ_FAILED -273 E_TAG_VALUE_OVER_RANGE -274 E_TAG_INVALID_LENGTH -278 E_TAG_NO_TAG -280 E_TAG_CANNOT_READ -281 E_TAG_READ_TIMEOUT GetCardUid UID を取得 GetCardData ﾃﾞｰﾀを取得

WriteRfidData

用途	GUI ﾎﾞﾀﾝをﾀｯﾌﾟして、RFID ﾀｸﾞにﾃﾞｰﾀを書き込みます。
書式	window.external.WriteRfidData(sData, sStartBlock, nWriteLen, nTimeout, nMode, sLoginKey, nLoginKeyType)
引数	**sData** 書き込みを行うﾃﾞｰﾀを指定します。 **nStartBlock** 書き込み開始位置を指定します。 -1 RFID ﾀｸﾞのﾃﾞｨﾌｫﾙﾄﾌﾞﾛｯｸから書き込みを開始します。 ﾀｸﾞﾀｲﾌﾟ / ｽﾀﾝﾀﾞｰﾄﾞ / ﾃﾞｨﾌｫﾙﾄ開始ﾌﾞﾛｯｸ Mifare / ISO 14443A / 4 SR176 / ISO 14443B / 4 ICODE SLI / ISO 15693 / 3 LRI512 / ISO 15693 / 0 SRF55VxxP / ISO 15693 / 3 EM4135 / ISO 15693 / 0 Tag-It / ISO 15693 / 0 その他 / ISO 15693 / 0 ICODE / Phillips / 5 **nWriteLen** 書き込みを行うﾊﾞｲﾄ数を指定します。RFID ﾀｸﾞの種類により、指定できる最大値は異なります。最大値を超える値が指定された場合は、自動的に最大値に調整されます。 **nTimeout** 予備(現在は使用していません) **nMode** 書き込みﾓｰﾄﾞを指定します。 0 直ちに書き込みを開始 0 以外 ﾄﾘｶﾞﾎﾞﾀﾝを押すと書き込みを開始 **sLoginKey** Mifare Standard 1K/4K や SLE66R35 など特定の RFID ﾀｸﾞへｱｸｾｽするためのﾛｸﾞｲﾝｷｰを 12 ﾊﾞｲﾄ長の 16 進数表記で指定します。 ﾛｸﾞｲﾝｷｰ例 / 16 進数表記文字列 0 / FFFFFFFFFFFF F1F2F3F4F5F6 / F1F2F3F4F5F6 **nLoginKeyType** ﾛｸﾞｲﾝｷｰﾀｲﾌﾟを指定します。 値 / ﾛｸﾞｲﾝｷｰﾀｲﾌﾟ 0 / Key A 1 / Key B

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	Var sData = “AABBCCDD”;
	Var nStartBlock = -1;
	Var nWriteLen = 8;
	Var nTimeout = 3;
	Var nMode = 0;
	Var sLoginKeyType = ‘FFFFFFFFFFFF’
	Var nLoginKeyType = 0;
	
	Var nWriteRfidData = window.external.WriteRfidData(sData,
	nStartBlock, nWriteLen, nTimeout, sLoginKey, nLoginKeyType);
	
	Alert(“WriteRfidUID=”+nWriteRfidData);
	...
	}
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、0 を返します。 ｴﾗｰの場合は、下記のｴﾗｰｺｰﾄﾞ(ﾏｲﾅｽ値)を返します。 -101 E_READER_NOT_INIT -273 E_TAG_VALUE_OVER_RANGE -274 E_TAG_INVALID_LENGTH -275 E_TAG_GET_DATA_FAILED -276 E_TAG_WRITE_FAILED -277 E_TAG_CANNOT_WRITE -278 E_TAG_NO_TAG -279 E_TAG_WRITE_TIMEOUT

WorkingType

用途	動作させる RFID 規格ﾀｲﾌﾟを設定します。 この関数を実行する前に、OpenCard 関数を実行する必要があります。
書式	window.external.WorkingType(nType)
引数	nType　　動作させる RFID 規格ﾀｲﾌﾟを指定します。 0x41　　ISO 14443A(ﾃﾞｨﾌｫﾙﾄ) 0x42　　ISO 14443B 0x31　　ISO 15693 0x73　　SR176/SRIX4K(ISO 14443B)

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　window.external.WorkingType(0x41);
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

AntennaControl

用途	ｱﾝﾃﾅを制御し、省電力運用を可能にします。 この関数を実行する前に、OpenCard 関数を実行する必要があります。
書式	window.external.AntennaControl(nSelect)
引数	nSelect　　ｱﾝﾃﾅｽﾃｰﾀｽを指定します。 0x00　　ｱﾝﾃﾅｵﾌ 0x01　　ｱﾝﾃﾅｵﾝ 0x38　　自動切り替え 0x10　　低消費ﾓｰﾄﾞ
ｺｰﾄﾞ例	<HTML> <BODY onload=”test()”> <SCRIPT> function test() { 　　window.external.AntennaControl(0x01) ;} … </SCRIPT> </BODY> </HTML>
備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

OpenCard

用途	ISO 14443A ｶｰﾄﾞを選択し、ｶｰﾄﾞ仕様情報を取得します。 ISO 14443A ｶｰﾄﾞを操作する前に、この関数を実行する必要があります。
書式	window.external.OpenCard()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	Var nOpenCard = window.external.OpenCard();
	Var sCardUid = window.external.GetCardUid();
	Var sCardAtqa = window.external.GetCardAtqa();
	Var sCardSak = window.external.GetCardSak();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardUid　UID を取得 GetCardAtqa　ATQA ｶｰﾄﾞﾀｲﾌﾟを取得 GetCardSak　SAK ｶｰﾄﾞﾀｲﾌﾟを取得

CloseCard

用途	ISO 14443A ｶｰﾄﾞをｸﾛｰｽﾞします。 ISO 14443A ｶｰﾄﾞへのﾃﾞｰﾀ書き込みを終了する場合に、この関数を実行します。以降、続けて読み取りを行うことができます。
書式	window.external.CloseCard()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	window.external.CloseCard();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

ReadMifareOneBlock

用途	ISO 14443A ｶｰﾄﾞから指定ﾌﾞﾛｯｸ位置のﾃﾞｰﾀを読み取ります。
書式	window.external.ReadMifareOneBlock(nKeyType, nBlock, sKey)
引数	nKeyType　ｷｰﾀｲﾌﾟを指定します。 値　定数　説明 0x00　CARD_KEY_A　ｷｰ A 0x01　CARD_KEY_B　ｷｰ B nBlock　読み取りを行うﾌﾞﾛｯｸ番号を指定します。 S50　ﾌﾞﾛｯｸ 0~63(ｾｷｭﾘﾃｨﾌﾞﾛｯｸを除く、計 45 ﾌﾞﾛｯｸ) S70　ﾌﾞﾛｯｸ 0~255(ｾｷｭﾘﾃｨﾌﾞﾛｯｸを除く、計 210 ﾌﾞﾛｯｸ) sKey　ｶｰﾄﾞｷｰを指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	var nKeyType = 0x00; // ｷｰ A
	var nBlock = 4;
	var sKey = ‘FFFFFFFFFFFF’;
	var sReadMifareOneBlock =
	window.external.ReadMifareOneBlock(nKeyType,
	nBlock, sKey);
	var sCardData = window.external.GetCardData();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardData　ｶｰﾄﾞﾃﾞｰﾀを取得

WriteMifareOneBlock

用途	ISO 14443A ｶｰﾄﾞの指定ﾌﾞﾛｯｸ位置にﾃﾞｰﾀを書き込みます。
書式	window.external.WriteMifareOneBlock(nKeyType, nBlock, sKey, sData)
引数	nKeyType　ｷｰﾀｲﾌﾟを指定します。 値　定数　説明 0x00　CARD_KEY_A　ｷｰ A 0x01　CARD_KEY_B　ｷｰ B nBlock　書き込みを行うﾌﾞﾛｯｸ番号を指定します。 S50　ﾌﾞﾛｯｸ 0~63(ｾｷｭﾘﾃｨﾌﾞﾛｯｸを除く、計 45 ﾌﾞﾛｯｸ) S70　ﾌﾞﾛｯｸ 0~255(ｾｷｭﾘﾃｨﾌﾞﾛｯｸを除く、計 210 ﾌﾞﾛｯｸ) sKey　ｶｰﾄﾞｷｰを指定します。 sData　書き込みﾃﾞｰﾀを指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	var nKeyType = 0x00; // ｷｰ A
	var nBlock = 4;
	var sKey = ‘FFFFFFFFFFFF’;
	var sData = ‘1234567890’;
	var sWriteMifareOneBlock =
	window.external.WriteMifareOneBlock(nKeyType,
	nBlock, sKey, sData);
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

ReadUltraLight

用途	Mifare Ultralight ｶｰﾄﾞから指定ﾌﾞﾛｯｸ位置のﾃﾞｰﾀを読み取ります。
書式	window.external.ReadUltraLight(nBlock)
引数	nBlock　　読み取りを行うﾌﾞﾛｯｸ番号を指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var nBlock = 0;
	　　var sReadUltraLight =
	　　window.external.UltraLight(nBlock);
	　　var sCardData = window.external.GetCardData();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardData　ｶｰﾄﾞﾃﾞｰﾀを取得

WriteUltraLight

用途	Mifare Ultralight ｶｰﾄﾞの指定ﾌﾞﾛｯｸ位置にﾃﾞｰﾀを書き込みます。
書式	window.external.WriteUltraLight(nBlock, sData)
引数	nBlock　　書き込みを行うﾌﾞﾛｯｸ番号を指定します。 sData　　書き込みﾃﾞｰﾀを指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var nBlock = 0;
	　　var sData = ‘1234’
	　　var sWriteUltraLight =
	　　window.external.WriteUltraLight(nBlock, sData);
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

STCardSelect

用途	ISO 14443B ｶｰﾄﾞを選択し、ｶｰﾄﾞ仕様情報を取得します。 ISO 14443B ｶｰﾄﾞを操作する前に、この関数を実行する必要があります。
書式	window.external.STCardSelect()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	Var nSTCardSelect = window.external.nSTCardSelect();
	Var sCardSelectNum = window.external.GetCardSelectNum();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A. 戻り値ﾘｽﾄ」を参照ください。 GetCardSelectNum　ｶｰﾄﾞ番号を取得

SR176ReadBlock

用途	SR176 ｶｰﾄﾞから指定ﾌﾞﾛｯｸ位置のﾃﾞｰﾀを読み取ります。
書式	window.external.SR176ReadBlock(nBlock)
引数	nBlock　読み取りを行うﾌﾞﾛｯｸ番号 0~15 を指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	var nBlock = 0;
	var sSR176ReadBlock =
	window.external.SR176ReadBlock(nBlock);
	var sCardData = window.external.GetCardData();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A. 戻り値ﾘｽﾄ」を参照ください。 GetCardData　ｶｰﾄﾞﾃﾞｰﾀを取得

SR176WriteBlock

用途	SR176 ｶｰﾄﾞ の指定ﾌﾞﾛｯｸ位置にﾃﾞｰﾀを書き込みます。
書式	window.external.SR176WriteBlock(nBlock, sData)
引数	nBlock　　書き込みを行うﾌﾞﾛｯｸ番号 4~14 を指定します。 sData　　書き込みﾃﾞｰﾀを指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var nBlock = 4;
	　　var sData = “12”;
	　　var sSR176WriteBlock =
	　　window.external.SR176WriteBlock(nBlock, sData);
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

SRIX4KReadBlock

用途	SRIX4K ｶｰﾄﾞ から指定ﾌﾞﾛｯｸ位置のﾃﾞｰﾀを読み取ります。
書式	window.external.SRIX4KReadBlock(nBlock)
引数	nBlock　　読み取りを行うﾌﾞﾛｯｸ番号 0~127 を指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var nBlock = 0;
	　　var sSR176ReadBlock =0;
	　　var sSRIX4KReadBlock =
	　　window.external.SRIX4KReadBlock(nBlock);
	　　var sCardData = window.external.GetCardData();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardData　ｶｰﾄﾞﾃﾞｰﾀを取得

SRIX4KWriteBlock

用途	SRIX4K ｶｰﾄﾞ の指定ﾌﾞﾛｯｸ位置にﾃﾞｰﾀを書き込みます。
書式	window.external.SRI4KWriteBlock(nBlock, sData)
引数	nBlock　　書き込みを行うﾌﾞﾛｯｸ番号 7~127 を指定します。 sData　　書き込みﾃﾞｰﾀを指定します。

ｺｰﾄﾞ 例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var nBlock = 0;
	　　var sData = “1234”;
	　　var nSRIX4KWriteBlock =
	　　window.external.SRIX4KWriteBlock(nBlock, sData);
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

SRIX4KReadUID

用途	SRIX4K ｶｰﾄﾞ から UID を読み取ります。
書式	window.external.SRIX4KReadUID()
引数	無し

ｺｰﾄﾞ 例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var sSRIX4KReadUID =
	　　window.external.SRIX4KReadUID();
	　　var sCardUid = window.external.GetCardUid();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardUid　　UID を取得

ISO15693Inventory

用途	ISO 15693 ｶｰﾄﾞを選択し、ｶｰﾄﾞをﾚﾃﾞｨﾓｰﾄﾞに設定して、UID を読み取ります。
書式	window.external.ISO15693Inventory()
引数	無し

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	var nInventory = window.external.ISO15693Inventory();
	var sCardMask = window.external.GetCardMask();
	var sCardDsfid = window.external.GetCardDsfid();
	var sCardUid = window.external.GetCardUid();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardMask　ﾏｽｸﾃﾞｰﾀを取得 GetCardDsfid　ﾃﾞｰﾀﾌｫｰﾏｯﾄ ID を取得 GetCardUid　UID を取得

ISO15693Read

用途	ISO 15693 ｶｰﾄﾞから指定ﾌﾞﾛｯｸ位置のﾃﾞｰﾀを読み取ります。
書式	window.external.ISO15693Read(sUID, nBlockStart, nBlockCount)
引数	sUID　読み取りを行うｶｰﾄﾞの UID を指定します。 nBlockStart　読み取りを行うﾌﾞﾛｯｸ開始位置を指定します。 nBlockCount　読み取りを行うﾌﾞﾛｯｸ数 1~15 を指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	var sUID = window.external.GetCardUid;
	var nBlockStart = 0;
	var nBlockCount = 1;
	var sISO15693Read =
	window.external.UltraLight(sUID, nBlockstart,
	nBlockCount);
	var sCardData = window.external.GetCardData();
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。 GetCardData　ｶｰﾄﾞﾃﾞｰﾀを取得

ISO15693Write

用途	ISO 15693 ｶｰﾄﾞの指定ﾌﾞﾛｯｸ位置にﾃﾞｰﾀを書き込みます。
書式	window.external.ISO15693Write(sUID, nBlock, sData)
引数	sUID　　書き込みを行うｶｰﾄﾞの UID を指定します。 nBlock　　書き込みを行うﾌﾞﾛｯｸ番号を指定します。 sData　　書き込みﾃﾞｰﾀを指定します。

ｺｰﾄﾞ例	<HTML>
	<BODY onload=”test()”>
	<SCRIPT>
	function test() {
	　　var sUID = window.external.GetCardUid;
	　　var nBlock = 0;
	　　var sData = ‘1234’
	　　var sISO15693Write =
	　　window.external.ISO15693Write(sUID, nBlock, sData);
	}
	…
	</SCRIPT>
	</BODY>
	</HTML>

備考	正常終了した場合は、戻り値として、LRSUCCESS を返します。 その他の戻り値に関しては、「補足 A．戻り値ﾘｽﾄ」を参照ください。

補足 A. 戻り値ﾘｽﾄ

名称	値	説明
LRSUCCESS	0x00	正常終了
LRSYSTEM	0x01	未知のｴﾗｰ
LRLASTCARD	0x02	前のｶｰﾄﾞがまだ存在
LRNOCARD	0x03	ｶｰﾄﾞが未存在
LRCTYPE	0x04	ｶｰﾄﾞﾀｲﾌﾟｴﾗｰ
LRPARAM	0x05	要求ﾊﾟﾗﾒｰﾀｴﾗｰ
LRACCESS	0x06	ｶｰﾄﾞｱｸｾｽｴﾗｰ
LRREAD	0x07	ｶｰﾄﾞﾘｰﾄﾞｴﾗｰ
LRWRITE	0x08	ｶｰﾄﾞﾗｲﾄｴﾗｰ
LRINCR	0x09	ﾊﾟｰｽｲﾝｸﾘﾒﾝﾄｴﾗｰ
LRDECR	0x0A	ﾊﾟｰｽﾃﾞｨｸﾘﾒﾝﾄｴﾗｰ
LRTRANSFER	0x0B	ﾊﾟｰｽﾊﾞﾘｭｰﾄﾗﾝｽﾌｧｰｴﾗｰ
LRRESTORE	0x0C	ﾊﾟｰｽﾘｽﾄｱｴﾗｰ
LRPURSE	0x0D	ﾊﾟｰｽﾊﾞﾘｭｰｶﾗﾌﾟﾄｴﾗｰ(改ざん)
LRMADERR	0x0E	ｶｰﾄﾞﾃﾞｨﾚｸﾄﾘｴﾗｰ
LRFIXERR	0x0F	ﾊﾟｰｽﾌｨｯｸｽｴﾗｰ
LRFIXED	0x10	ﾊﾟｰｽﾊﾞﾘｭｰｶﾗﾌﾟﾄ(改ざん)検知･修正済み
LRNOTOPEN	0x11	ｶｰﾄﾞがｵｰﾌﾟﾝされていない
LRNOFILE	0x12	ﾌｧｲﾙが見つかりません
LRBADSIZE	0x13	ﾌｧｲﾙｻｲｽﾞｴﾗｰ
LAABORTED	0x14	ﾘｸｽﾄがｱﾎﾞｰﾄしました
LRMANYCARD	0x15	ｶｰﾄﾞが多く存在し過ぎです
LRFORMAT	0x16	ｶｰﾄﾞﾌｫｰﾏｯﾄｴﾗｰ
LRCREATE	0x17	ｶｰﾄﾞﾌｧｲﾙｸﾘｴｲﾄｴﾗｰ
LRDELETE	0x18	ｶｰﾄﾞﾌｧｲﾙ削除ｴﾗｰ
LRALREADOPEN	0x19	ｶｰﾄﾞは既にｵｰﾌﾟﾝされています
LRALREADCLOSED	0x1A	ｶｰﾄﾞは既にｸﾛｰｽﾞされています
LRMSTRKEYLOAD	0x1B	ﾏｽﾀｰｷｰをﾛｰﾄﾞできません
LRAPPKEYLOAD	0x1C	ｱﾌﾟﾘｹｰｼｮﾝｷｰをﾛｰﾄﾞできません
LRKEYCARD	0x1D	ｷｰｶｰﾄﾞｴﾗｰ
LRUNFORMAT	0x1E	ｶｰﾄﾞにﾌｧｲﾙが存在します
LRNOKBDCHAR	0x20	ｷｰﾎﾞｰﾄﾞｷｬﾗｸﾀではありません
LRAPIWRITE	0x21	API ﾃﾞｰﾀﾗｲﾄｴﾗｰ
LRAPIREAD	0x22	API ﾘｰﾄﾞｴﾗｰ
LRBLOCKERROR	0x23	API ﾃﾞｰﾀﾘｰﾄﾞｴﾗｰ
LRNOTIMPR	0x7F	無効な関数です
LRUNKNOWN	0x80	未知のｴﾗｰ
LRCCRBUSY	0xBB	ﾘｰﾀﾞがﾋﾞｼﾞｰ状態です
LRCHANGEWROKTYPE	0xEE	ﾘｰﾀﾞﾜｰｷﾝｸﾞﾀｲﾌﾟ切り替えｴﾗｰ
LRRDUNKNOWN	0xEF	未知のﾘｰﾀﾞｴﾗｰ
LRCRDNOTOPEN	0xFA	ｶｰﾄﾞがｵｰﾌﾟﾝされていません
LRINUSE	0xFB	他のｱﾌﾟﾘｹｰｼｮﾝがｶｰﾄﾞ使用中です
LRAPPLICERR	0xFC	API ｼｽﾃﾑｴﾗｰ
LRRDLINKLOS	0xFD	ﾘｰﾀﾞとのﾘﾝｸに失敗しました
LRBADCOMPORT	0xFE	COM ﾎﾟｰﾄにｱｸｾｽできません
LRRDNOINIT	0xFF	ﾘｰﾀﾞがｵｰﾌﾟﾝされてません
LRNOCRYPTBOX	0xF9	ｷｰﾎﾞｯｸｽが見つかりません
LRBADAPPACCESS	0xF8	無効なｱﾌﾟﾘｹｰｼｮﾝｱｸｾｽｺｰﾄﾞです
LRBADRDACCESS	0xF7	無効なﾘｰﾀﾞｱｸｾｽｺｰﾄﾞです
LRNOMAIDFILE	0xF6	MAID 定義ﾌｧｲﾙをｵｰﾌﾟﾝできません
LRBOXREAD	0xF5	ｷｰﾎﾞｯｸｽからﾘｰﾄﾞできません
LRBOXWRITE	0xF4	ｷｰﾎﾞｰﾄﾞﾎﾞｯｸｽにﾗｲﾄできません
LRBOXNOKEYS	0xF3	ﾎﾞｯｸｽのｷｰが無い又は無効です
LRSECURE	0xF2	MAC 検査異常
LRERRSELREADER	0xF1	指定ﾘｰﾀﾞに切り替えできません
LRRDNORESP	0xF0	ﾘｰﾀﾞ応答なし