آموزش برنامه نویسی برای موبایل

MIDLETPASCAL

بهمن ماه 1391

lord_viper
AHA [godvb]

فهرست

Midletpascal چیست ؟

Midletpascal یک کامپایلرو محیط توسعه (IDE) مخصوص ایجاد ابزارهای کاربردی موبایل تهیه شده است که کامپایلرآن کد نوشته شده به زبان پاسکال را به بایت کد جاوا تبدیل می کند. پس اگر با یکی ازنسخه های زبان برنامه نویسی پاسکال مانند لازاروس یا دلفی آشنایی داشته باشید براحتی می توانید برای گوشی های همراه خود برنامه بنویسید .برنامه هایی که با این کامپایلر ایجاد می شوند قابلیت اجرا بر روی هر نوع وسیله سیار و قابل حمل همچون موبایل را دارند. امروزه اکثر برنامه هایی که برای موبایل نوشته می شوند توسط زبان جاوا ایجاد می شوند ، ولی با استفاده از این کامپایلر شما هیچ احتیاجی به دانستن زبان جاوا ندارید و اگر با پاسکال یا دلفی آشنایی دارید می توانید کار خود با این نرم افزار را شروع کنید. این کامپایلر همراه یک محیط کاربر پسند برای ویندوز عرضه شده است. مزیت Midletpascal این است که کد نوشته شده را مستقیما به یک low-level کد جا وا تبدیل می کند که قابلیت اجرا روی هر دستگاه java me دارد.کدهای تولید شده توسط این کامپایلر برای اجرا نیازی به کامپایلر جاوا ندارد . در اینترنت ابزارهای مشابهی وجود دارد ولی کدهایی که آنها برای شما تولید می کنند یک کد واسط (intermediate-code) می باشد که با یک مترجم آن را به یک فایل jar تبدیل می کنند که برای اجرا نیاز به کامپایلر جاوا دارید ، در حالی که Midletpascal شما را از این امور بی نیاز می کند.این IDE در ویندوز استفاده می گردد هرچند که می توانید در سیستم عاملهای Linux and MacOSدر Wine اجرا نمایید.برنامه های نوشته شده تحت این IDE در تمامی موبایل ها اجرا می شوند و از پلتفرم های MIDP 1.0 and CLDC 1.0 نیز پشتیبانی میکند.نویسنده این محیط توسعه Niksa Orlic (norlic) میباشد که سورس برنامه MIDletPascal 2.02 را در سپتامبر 2009 تحت لیسانس GNU پابلیک کرد و در اختیار عموم قرار داد.که در اکتبر 2009 ورژن3 این IDE استارت خورد و روز به روز به امکانات وقابلیت های آن افزوده شد که در حال حاظر اخرین نسخه این IDE 3.5 میباشد.کتابخانه های زیادی برای این ide نوشته شده که قابلیتهای مختلفی در اختیار شما قرار میدهد و حتی صفحه نمایشهای لمسی را پشتیبانی میکنند. آخرین ورژن این محیط برنامه نویسی را می توانید از آدرس زیر دریافت کنید:

http://sourceforge.net/projects/midletpascal/

فروم

http://sourceforge.net/projects/midletpascal/forums/forum/1013750/index/page/1

ویکی پدیا

http://en.wikipedia.org/wiki/MIDletPascal

مزایای Midletpascal

1.ایجاد بایت کدهای سطح پایین-کوچک و سریع برای جاوا

2.پشتیبانی کامل از قواعد پاسکال

3.بخشی از کد را میتوانید مستقیما در جاوا بنویسید

4.sms messaging

5.ارتباط با پروتکل http

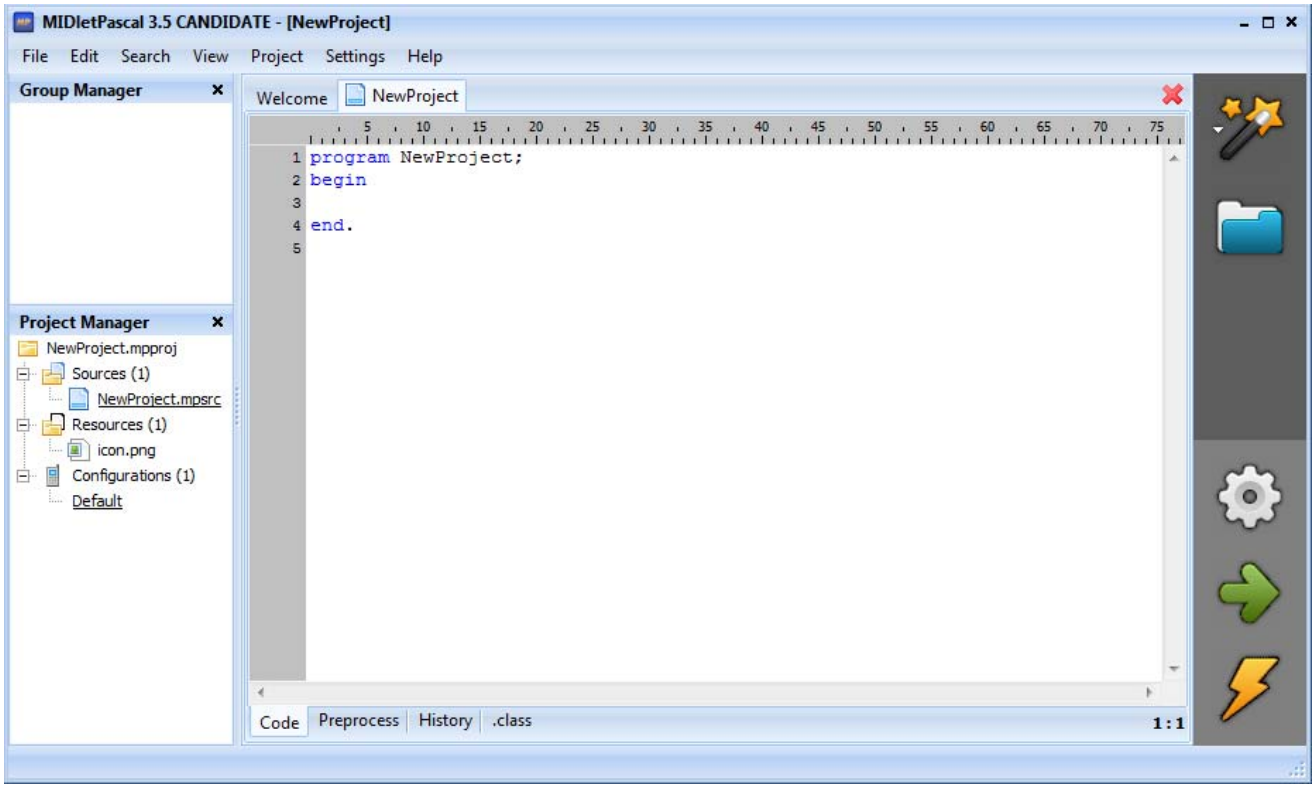
6.پشتیبانی از رابط کاربری و کنترلها User Interface

7.پشتیبانی از چند رسانه ای

8.دارای محیط توسعه کاربر پسند

9.همچنان در حال توسعه میباشد

آشنایی با محیط و ساختار برنامه نویسی در Midletpascal



ساختار کلی برنامه نویسی در این محیط بدین صورت است که بدن اصلی و بقولی تابع main مان در مکان statementsقرار دارد.که قبل از آن می توان type ها،متغیرها و دیگر پروسیجر مورد نیازمان را تعریف نماییم

```
program programName;
type
definitions;
constant
declarations;
variable
declarations;
function and procedure declarations;
begin
  statements
end.
```

این محیط برنامه نویسی case-insensitive است یعنی بر حروف کوچک و بزرگ حساس نیست یعنی بین begin و BEGIN تفاوتی قایل نمیشود

برای کامنت های چند خطی می توانید متن خود را درون '{' and '}' قرار دهید و یا: (* and *)

(* this is also a comment *)
{ this is a comment }

همچنین میتوان از استایل زبان C برای کامنت گذاری استفاده نمود:

// this is a single-line comment
/* this is also a multi-line comment */

Type definitions: می بایست Type ها و یا همان ساختاری که مورد نیازتان هست را تعریف نمایید که حتما باید از کلمه کلیدی type نیز استفاده نمایید مانند زیر:

```
program myOwnTypes;
  type
number = integer;
chessFieldType = array[1..8, 1..8] of integer;
 chessElementType = record
     positionRow, positionCol: integer;
     elementType: integer;
end;
  var
element: chessElementType;
field: chessFieldType;
numberOfPlayers: number;
```

```
begin
end.
```

constant declarations: در این بخش شما می بایست ثابت ها را همراه با نام آنها ومقادیرشان را بنویسید.در مثال زیر می توانید مشاهد کنید:

```
program constantDeclarations;
const
minutesInHour = 60;
 hoursInDay = 24;
 famousQuote = 'To be or not to be';
begin

end.
```

variable declarations: در این بخش شما متغییرهایتان را تعریف مینمایید که می بایست برای تعریف آنها از کلمه کلیدی varاستفاده نمایید:

```
program variableDeclarations;
var
 index: integer;
 field: array [1..15] of integer;
begin
 for index := 1 to 15 do
  field[index] := 10;
end.
```

عملگر انتساب نیز به شکل زیر است که از := استفاده میشود:

```
variable := value;
```

فراخوانی یک روتین تنها نام آن را ذکرکرد:

```
routine;
```

Data type در MIDLETPASCAL

Data typeها را بطور کلی به دو دسته ساده و مرکب تقسیم میکنیم.

ساده عبارتند از:

نوع Boolean فقط می تواند دو مقدار true or false را به خود اختصاص دهد.عملگرهایی که برروی این نوع می توان انجام داد به شرح زیر است:

- عملگرهای مقایسه ای<> ,= :
- عملگرهای منطقیor, xor
- عملگرand
- عملگرnot

عملگر not داراری بالاترین الویت و <> and = پایین ترین الویت است.بعنوان مثال عبارت زیر true است:

((true and false) = false)

نوع Char که اشاره به یک کاراکتر دارد.عملگرهایی که برروی این نوع انجام میشود:

- عملگرهایی مقایسه ای =< ,=> ,< ,> ,<> ,=که مقدار اسکی دو کاراکتر را مورد بررسی قرار میدهد.
- الحاق رشته ها + ، که رشته دومی را در انتهای رشته اول میچسباند.

بازهم یک مثال که عبارت زیر false است:

('a' = 'A')

برای گرفتن کد اسکی یک کاراکتر از تابع ord استفاده میشود و برعکس برای تبدیل کداسکی به کاراکتر معادل از تابع chr استفاده میکنیم که در ادامه توضیح داده خواهند شد.

نوع Integer که می تواند مقدار عددی در رنج -2,147,483,647 2,147,483,647 and را به ان نسبت داد.عملگرهای مورد استفاده برروی این نوع:

- عملگرهای مقایسه ای =< ,=> ,< ,> ,<> ,=
- عملگرهای جمع و تفریق- ,+
- عملگرهای ضرب و تقسیم(div, mod ,/ ,* هیچ تفاوتی بین / و divوجود ندارد)
- عملگرهای شیفتshl, shr

نوع String برای متون استفاده میشود.در دلفی String را میتوان بعنوان آرایه ای از کاراکترها در نظر گرفت و با آن بصورت آرایه رفتار کرد ولی در MIDletPascal اینطور نیست.

- عملگرهای مقایسه ای(<> ,= به کارکترهای بزرگ و کوچک حساس است)
- عملگرهای الحاقی+

به مثال زیر توجه کنید :

```
var text: string;
 begin
   text := 'It is now ' + getHour(getCurrentTime) + ' o"clock ';
   drawText(text, 0, 0);
   repaint;
   delay(2000);
 end.
```

در این مثال ما زمان را نمایش میدهیم.میبینید که رشته در (single-quote) ' و برای نمایش خود کاراکتر single-quoteآن را دوبار تکرار کرده است.

الحاق رشته ها به همدیگر بصورت هوشمند.یعنی دیگراحتیاجی به استفاده از عملگر "+" نیست مانند زیر:

```
s := 'this is'#10'a test';
```

نوع داده ای Image در حافظه قرار دارد که میتوان توسط تابع loadImage عکسی را در متغیری از این نوع لود کرد.

نوع real برا نمایش اعداد حقیقی استفاده میشوند.اعمالی که می توان برروی این نوع داده انجام داد:

- اعمال مقایسه ای=< ,=> ,< ,> ,<> ,=
- اعمال جمع و تفریق- ,+
- اعمال ضرب و تقسیم/ ,*

نوع Record

در این نوع شما می توانید مجموع از اوناع متغیرها را داشته باشید.بعنوان مثال شما یک بازی نوشته اید که قهرمان این بازی را کاربر کنترل می کند پس شما باید یک record تعریف نمایید که مختصات آن را درصفحه و مقدار جان باقی مانده را در خود نگه دارد.مانند:

```
type
heroType = record
    positionX, positionY: integer;
    health: integer;
 end;
```

برای ساختن یک نمونه از record بالا بدین صورت می نویسیم:

```
var
hero: heroType;
```

و همچنین می توانید پارامترهای ورودی یک روتین را از این نوع تعریف نمایید:

```
function isHeroDead(hero: heroType): boolean;
 begin
 if (hero.health <= 0) then
    isHeroDead := true;
```

```
  else
    isHeroDead := false;
end;
```

recordنیز می تواند بجای کلمه کلیدی type از var نیز استفاده نمایید:

```
var
hero: record;
      begin
           positionX, positionY: integer;
           health: integer;
       end;
```

با قرار دادن dot می توانید به داده های record تعریف شده دسترسی پیدا کنید:

```
...
 { move the hero to the right }
 hero.positionX := hero.positionX + 1;
 ...
```

و نکته دیگر در مورد این نوع داده ای که شما نمی توانید مستقیما دونوع متفاوت تعریف شده از record درون هم کپی کنید و یا انتساب دهید.بلکه می بایست مقدارهای داخلی آنها را کپی نمایید.

```
var a, b: record
           x: integer;
        end;
 begin
   ...
   a := b; { this is not valid }
   ...
   a.x := b.x; { copy element by element instead }
 end.
```

نوع Array

برای تعریف متغییرها ما به این صورت عمل میکردیم

```
var
a:integer;
```

با این کار مقداری از حافظه برای a رزرو میشد تا مقدار عددی در ان قرار گیرد اما اگر بخواهیم 10 عدد را نگهداری کنیم چه؟ تعریف 10 متغییر؟ اگر بخواهیم 1000 عدد را نگهداری کنیم چه؟ استفاده از ارایه به ما این امکان را میدهد تا برای 1000 متغییر 1 نام تعریف کرده و با استفاده از شماره ان عنصر به ان دستیابی پیدا کنیم اعضای یک ارایه در حافظه در مجاور هم قرار میگیرند و به اسانی میتوان در هر قسمت اطلاعات را وارد کرد یا خواند.آرایه ها بدین صورت تعریف میشوند:

```
type
chessFieldType = array[1..8, 1..8] of integer;
```

برای تعریف یک آرایه 8*8 که یک آرایه دوبعدی میشود بدین صورت انجام میشود:

```
type
  chessFieldType = array[1..8, 1..8] of integer;
var
  chessField: chessFieldType;
```

که برای ساده ترشدن دوخط کد بالا معادل تکه کد زیر است:

```
var
chessField: array[1..8, 1..8] of integer;
```

در مثال زیر خانه های خالی از جدول شطرنج مان را میشماریم:

```
type
chessFieldType = array[1..8, 1..8] of integer;
 var
chessField: chessFieldType;
   i, j, count: integer;
 begin
  ...
  { initialize the chess field to contain some elements }
  ...
  count := 0;
  for i:=1 to 8 do
    for j:=1 to 8 do
      if chessField[i, j] = 0 then count := count + 1;
 end.
```

می توان آرایه های چند بعدی داشته باشید ولی به این نکته توجه کنید که آرایه های بعد زیاد حافظه زیادی را اشغال می کنند.توجه داشته باشید حداکثر طول ارایه ها برابر با (15 bits) 32767 میباشد.

حلقه ها

دستور for

Syntaxاین دستور بصورت زیر است:

```
for loopIndex := initialValueExpression to finalValueExpression do
begin
 statements
end;
```

بجای to می توانید از downto استفاده که مقدار شمارشگر حلقه را کاهش میدهد.تکه کد زیر جمع اعداد از 1 تا 10 را محاسبه می کند:

کد:

```
for i:= 1 to 10 do
 begin
  sum := sum + i;
 end;
```

دستور while نیز یک نوع دیگر ازحلقه ها محسوب میشود و syntax آن نیز بصورت زیر است:

```
while condition do
begin
 statements
end;
```

برای منتظرماندن ورودی کاربر می توان این چنین بنویسیم:

```
while (getKeyClicked = KE_NONE) do
 begin
  delay(100);
 end;
```

دستور repeat/until نیزبدین صورت است:

```
repeat
 statements;
until endingCondition;
```

مثال قبلی را با این دستور میتوان بدین صورت پیاده سازی کرد:

```
repeat
  delay(100);
  until (getKeyClicked <> KE_NONE)
```

برای ایجاد یک حلقه بینهایت می توانید از دستورات REPEAT/FOREVER نیز استفاده نمایید.مثال:

```
repeat
 ...
 if condition then
```

```
  break;
 ...
forever;
```

دستور Break برای خروج از حلقه استفاده می شود که در تمامی انواع حلقه های ذکرشده بالا می توان استفاده نمود.به این مثال توجه کنید:

```
repeat
  for i := 1 to 10 do
  begin
    if doSomething(i) = -1 then break; // break from for-loop
  end;

 until getClickedCommand <> emptyCommand;
```

دستورات If-then-else

شکل کلی دستور بصورت زیر است:

```
if condition then
begin
 statements; { condition true branch }

end
[
else
begin
 statements; { condition false branch }
end;
]
```

توجه:هرگاه دستورات بیش از یک خط بشود احتیاج begin/end خواهیم داشت.اگر دستور یک خط باشد می توان از begin/end استفاده نکرد.

شیفت دادن

استفاده از عمل های شیفت مانند SHL و SHR مانند زیر:

```
a := $A714;
  b := a shl 3;
  c := b shr 2;
```

همانند زبان C-C++ میتوانید از علامت های <>نیز برای شیفت دادن استفاده نمایید:

```
a := $714A;
  b := a << 3;
  c := b >> 2;
```

برای انجام عملیات شیفت به راست بدون علامت می توانید بصورت زیر عمل نمایید:

```
a := $2704;
  b := a ushr 1;
  c := b >>> 2;
```

نوع داده ایCommand

که برای ساخت منو ها می باشد که یک مثال در زیر امده و در ادامه بیشتر توضیح داده خواهد شد.

```
  var formCommand, canvasCommand: command;
  begin
    formCommand := createCommand('Show canvas', CM_SCREEN, 1);
    canvasCommand := createCommand('Show form', CM_SCREEN, 1);

    { add the canvas form onto the canvas (the canvas is displayed by default) }
    addCommand(canvasCommand);

    { switch to form display }
    showForm;
```

```
  { add the form command to the form display }
  addCommand(formCommand);

  { forever switch between the canvas and the form display }
  while true do
  begin
    if getClickedCommand = formCommand then
      showCanvas;
    if getClickedCommand = canvasCommand then
      showForm;
  end;
end.
```

نوع داده ایRecordStore

Recordstore همان فایل در سیستم خودمان محسوب میشود. برنامه ها تنظیمات و داده های موردنیازشان در درrecordstore ذخیره میکنند که در اجراهای بعد می توانند آنهارا بازیابی نمایند.هر برنامه فقط یک recordstoreدارد که ساختار آن شبیه آرایه است و دارای index است برای برنامه های موبایل که اکثرا کرک میکنم و چندتایی رو روی فروم قرار دادم از record برای ذخیره سریال و چک کردن استفاده میکنند که بهترین سرنخ برای رسیدن به هدف میباشد.

این تابع اطلاعات را در رکورد استورrs ریخته و ایندکس انرا برمیگرداند و در صورت بروز خطا 1- برمیگرداند

```
function addRecordStoreEntry(rs: recordStore; data: string): integer;
```

وقتی یک رکورداستور را باز میکنیم بعد از اتمام کارمان باید انرا ببندیم که از تابع زیر استفاده میکنیم

```
procedure closeRecordStore(rs: recordStore);
```

برای حذف یک دیتااستور با استفاده از نام ان از دستور زیر استفاده میشود

procedure deleteRecordStore(name: string);

مقدار یک data را با استفاده از ایندکس آن در یک دیتا استور حذف میکند

procedure deleteRecordStoreEntry(rs: recordStore; index: integer);

این تابع اندیس بعدی که تابع AddRecordStoreEntry اختصاص داده میشود را برمیگرداند

function getRecordStoreNextId(rs: recordStore): integer;

تعداد رکوردهای موجود در دیتااستور را برمیگرداند

function getRecordStoreSize(rs: recordStore): integer;

برای تغییر یا بروز رسانی یک رکورد در دیتا استور بر اساس ایندکس آن

procedure modifyRecordStoreEntry(rs: recordStore; newData: string; index: integer);

باز کردن یک دیتا استور بر اساس نام آن اگر وجود نداشته باشد یک دیتااستور خالی ساخته میشود

function openRecordStore(name: string): recordStore;

خواندن رکورد ذخیره شده در دیتا استور بر اساس ایندکس آن

function readRecordStoreEntry(rs: recordStore; index: integer): string;

مثال:

```
var rs: recordStore;
   index: integer;
   name: string;
begin
  // write some data inside record store
  rs := openRecordStore('names');
  index := addRecordStoreEntry(rs, 'John Smith');
  closeRecordStore(rs);

  // read the data
  rs := openRecordStore('names');
  name := readRecordStoreEntry(rs, index);
  closeRecordStore(rs);
end.
```

کار با http

نوع داده ای Http با MIDletPascal می توان یک ارتباط کامل HTTPرا داشت.در یک ارتباط Http می بایست پارامترهایی را مقداردهی کنید.که ترتیب عملیات یک ارتباط به شرح زیر است:(life cycle)

1. ایجاد ارتباط با وب سرور

```
function openHttp(httpConn: http; url: string):boolean;
```

2. چک کردن باز بودن ارتباط با سرور

```
function isHttpOpen(httpConn: http):boolean;
```

3. تنظیم متد ارسال درخواست که میتواند مقادیری برابر با GET , POST , HEAD داشته باشد

```
procedure setHttpMethod(httpConn: http; method:string);
```

4. افزودن فیلد هدر درخواست (اختیاری)

```
procedure addHttpHeader(httpConn: http; name, value:string);
```

5. افزودن بدنه درخواست که فقط در متد POST مورد نیاز است

```
procedure addHttpBody(httpConn: http; data: string);
```

6. ارسال درخواست و منتظر جواب بودن از وب سرور

```
function sendHttpMessage(httpConn: http): integer;
```

7. خواند فیلد هدر پاسخ

```
function getHttpHeader(httpConn: http; name: string): string;
```

8. خواندن محتوای پاسخ

```
function getHttpResponse(httpConn: http):string;
```

9. بستن ارتباط

```
procedure closeHttp(httpConn: http);
```

یک نمونه سورس که می توانید برای درک بهتر مسئله در زیر ببنید:

```
var conn: http;
    htmlBody: string;
    contentType: string;
 begin
  if not openHttp(conn, 'http://www.google.com') then halt;
  setHttpMethod(conn, GET);
  addHttpHeader(conn, 'User-agent', 'MIDletPascal browser');
```

```
  if sendHttpMessage(conn) <> 200 then halt;
  htmlBody := getHttpResponse(conn);
  contentType := getHttpHeader(conn, 'Content-type');
  closeHttp(conn);
end.
```

نوع داده Resource

نوع Resource که می توانید فایل هایی را بعنوان ریسورس در برنامه داشته باشید و درهنگام اجرا از آنها استفاده نمایید. با دادن نام یک ریسورس انرا باز میکند

```
function openResource(name: string): resource;
```

ریسورس مورد نظر را میبندد

```
procedure closeResource(res: resource);
```

بایت بعدی در ریسورس را میخواند اگر چیزی وجود نداشته باشد مقدار EOF یا ارور برمیگرداند

```
function readByte(res: resource):integer;
```

خط بعدی در ریسورس را میخواند اگر چیزی وجود نداشته باشد رشته برگردانده شده مقدار خالی یا ارور برمیگرداند

```
function readLine(res: resource):string;
```

اگر ریسورس مورد نظر وجود داشته باشد مقدار درستی و در غیر این صورت مقدار نادرستی برمیگرداند

```
function resourceAvailable(res: resource):boolean;
```

مثال:

```
var res   : resource;
    byte  : integer;
    line  : string;
    index : integer;
```

```
begin
    res := openResource('/data.txt');

    if (resourceAvailable(res)) then
    begin
        byte := readByte(res);
            line := readLine(res);
        closeResource(res);
    end;

    showForm;
    index := formAddString('Byte is: ' + chr(byte));
    index := formAddString('Line is: ' + line);
    delay(1000);
end.
```

توابع فرم

procedure addCommand(cmd: command);

این تابع برای اضافه کردن یک کامند به صفحه نمایش دستگاه استفاده میشود. توجه داشته باشبد بعضی از موبایلها مانند موتورلا بدون فراخوانی تابع repaintبعد از این تابع کامند را ایجاد نمیکنند پس بعد از استفاده از تابع addCommand حتما از تابع repaint استفاده کنید

function choiceAppendString(choiceID: integer; itemText:string):integer;

اضافه کردن متن به یک choice group با استفاده از id آن

```
var choiceGroupID: integer;
   NY, LA: integer;
begin
  showForm;
  choiceGroupID := formAddChoice('Where do you live?', CH_EXCLUSIVE);
  NY := choiceAppendString(choiceGroupID, 'New York');
  LA := choiceAppendString(choiceGroupID, 'Los Angeles');
end.
```

```
function choiceAppendStringImage(choiceID: integer; itemText:string; img:image):integer;
```

اضافه کردن یک متن و عکس به یک choice group با استفاده از id آن

```
var choiceGroupID: integer;
   NY, LA: integer;
begin
  showForm;
  choiceGroupID := formAddChoice('Where do you live?', CH_EXCLUSIVE);
  NY := choiceAppendStringImage(choiceGroupID, 'New York', loadImage('/NY.png'));
  LA := choiceAppendStringImage(choiceGroupID, 'Los Angeles', loadImage('/LA.png'));
end.
```

```
function choiceGetSelectedIndex(choiceID: integer):integer;
```

ایندکس ایتم انتخاب شده در choice group را برمیگرداند اگر ایتمی انتخاب نشده باشد مقدار 1- را برمیگرداند

```
function choiceIsSelected(choiceID: integer; itemIndex:integer):boolean;
```

اگر ایتم مورد نظر در choice group انتخاب شده باشد مقدار درستی در غیر این صورت مقدار نادرستی برمیگرداند

```
var choiceGroupID: integer;
```

```
  NY, LA: integer;
begin
  showForm;
  choiceGroupID := formAddChoice('Where do you live?', CH_EXCLUSIVE);
  NY := choiceAppendStringImage(choiceGroupID, 'New York', loadImage('/NY.png'));
  LA := choiceAppendStringImage(choiceGroupID, 'Los Angeles', loadImage('/LA.png'));
  if choiceIsSelected(choiceGroupID, NY) then
    formAddString('New York');
  else
    formAddString('Los Angeles');
end.
```

```
procedure clearForm;
```

تمام المانها و کامندهای روی یک فورم را حذف میکند

```
var label_id, textField_id: integer;
begin
  label_id := formAddString('Hello world');
  textField_id := formAddTextField('Enter your name', 'Mr.Smith', 20, TF_ANY);
  showForm;
  delay(2000);
  clearForm;
  delay(2000);
end.
```

```
function createCommand(label:string; commandType:integer; priority:integer): command;
```

از این تابع برای ایجاد یک کامند استفاده میشود

توجه داشته باشید label کامند باید کوتاه باشد حق تقدم کامند بر اساس priority مشخص میشود عدد پایینتر یعنی تقدم بالا و عدد بالاتر یعنی تقدم پایینتر و مقدار commandType نوع کامند را مشخص میکند که میتواند مقادیر زیر باشد

```
  CM_SCREEN - any command type
  CM_BACK
  CM_CANCEL
  CM_OK
  CM_HELP
  CM_STOP
  CM_EXIT
  CM_ITEM

 var exitCmd, pauseCmd: command;
 begin
  exitCmd := createCommand('Exit', CM_EXIT, 1);
  pauseCmd := createCommand('Pause', CM_SCREEN, 1);
  addCommand(exitCmd);
  addCommand(pauseCmd);
 end.

function emptyCommand: command;
```

این تابع مقدار nonclicked را برمیگرداند واز این تابع معمولا برای چک کردن کلیک شدن یک کامند مورد استفاده قرار میگیرد

```
 var ok, clicked: command;
 begin
  ok := createCommand('OK', CM_OK, 1);
  addCommand(ok);
  repeat
    clicked := getClickedCommand;
  until clicked <> emptyCommand;
  if clicked = ok then halt else doSomething...
 end.
```

function formAddChoice(label:string; choiceType:integer):integer;

این تابع یک choice group به فورم اضافه میکند و id ان choice group را برمیگرداند .مقدار choiceType میتواند 2 مقدار زیر را داشته باشد

CH_EXCLUSIVE:1 ایتم فقط انتخاب شود

CH_MULTIPLE: تعداد دلخواه ایتم میتواند انتخاب شود

function formAddDateField(label: string; type:integer): integer;

اضافه کردن فیلد تاریخ و زمان به فورم مقدار type میتواند مقادیر زیر را داشته باشد

DF_DATE – فقط تاریخ

DF_TIME – فقط زمان

DF_DATE_TIME – نمایش تاریخ و زمان با هم

function formAddGauge(label:string; isInteractive:boolean; maxValue, initialValue:integer): integer;

اضافه کردن یک Gauge به فورم و مقدار id آن Gauge را برمیگرداند.توجه داشته باشید اگر مقدار isInteractive را برابر False قرار دهید دریگر مقدار Gauge را نمیتوانید تغییر دهید

function formAddImage(i:image):integer;

اضافه کردن یک کنترل عکس به فورم و id کنترل را برمیگرداند

```
var image_id: integer;
begin
  image_id := formAddImage(loadImage('/logo.png'));
```

```
  showForm;
  delay(2000);
end.
```

```
function formAddSpace:integer;
```

برای ایجاد فاصله بین کنترلهای روی فورم مورد استفاده قرار میگیرد و id کنترل را برمیگرداند

```
var label_id, space_id, textField_id: integer;
begin
  label_id := formAddString('Hello world');
  space_id := formAddSpace();
  textField_id := formAddTextField('Enter your name', 'Mr.Smith', 20, TF_ANY);
  showForm;
  delay(2000);
end.
```

```
function formAddString(s:string):integer;
```

برای ایجاد یک label (متن غیر قابل تغییر)روی فورم بکار میرود و id انرا برمیگرداند

```
var label_id: integer;
begin
  label_id := formAddString('Hello world');
  showForm;
  delay(2000);
end.
```

```
function formAddTextField(prompt, defaultValue: string; maxSize: integer; constraints:integer):
integer;
```

این تابع یک text field روی فورم ایجاد کرده و id انرا برمیگرداند.Prompt مقداری که بعدا در تکست فیلد نمایش داده میشود defaultValue مقدار پیش فرض maxSize حداکثر طول کاراکترها که در تکست فیلد قرار میگیرد constraints که میتواند دارای مقادیر زیر باشد

TF_ANY – هر کاراکتری را قبول میکند

TF_EMAIL – فقط ادرس ایمیل میتوان در ان وارد کرد

TF_NUMERIC – فقط میتوان عدد در ان وارد کرد

TF_PHONENUMBER – فقط میتوان شماره تماس در آن وارد کرد

TF_URL – فقط میتوان ادرس اینترنتی در آن وارد کرد

```
var textField_id: integer;
begin
  textField_id := formAddTextField('Enter your name', 'Mr.Smith', 20, TF_ANY);
  showForm;
  delay(2000);
end.
```

```
function formGetDate(index: integer): integer;
```

مقدار ساعت و تاریخ را در یک date filed قرار میدهد که یک مقدار عددی بر حسب ثانیه از 1 ژانویه 1970 میباشد که باید id آن date filed را به این تابع ارسال کنید

```
function formGetText(textFieldID:integer):string;
```

متن درون یک تکست فیلد را برمیگرداند.توجه داشته باشید که id تکست فیلد مورد نظر را باید به این تابع ارسال کنید

```
function formGetValue(gaugeID:integer):integer;
```

مقدار یک gauge را برمیگرداند . توجه داشته باشید که id آن gauge مورد نظر را باید به این تابع ارسال کنید

procedure formRemove(item:integer);

کنترل مورد نظر را از فورم حذف میکند. توجه داشته باشید که id کنترل مورد نظر را باید به این تابع ارسال کنید

procedure formSetDate(index: integer; dateTime: integer);

مقدار درون یک date filed را میتوان با این تابع تغییر داد.توجه داشته باشید اگر نوع date filed برابر با DF_TIME باشد باید مقدار dateاز DateTime آن برابر با 0 باشد

procedure formSetText(textFieldID:integer; text:string);

تغییر متن یک تکست فیلد با استفاده از ایدی آن

procedure formSetValue(gaugeID:integer; value:integer);

تغییر مقدار یک gauge با استفاده از ایدی آن

function getClickedCommand: command;

اخرین کامندی که کلیک شده را برمیگرداند

```
var exitCmd, clicked: command;
begin
  exitCmd := createCommand('Exit', CM_EXIT, 1);
  addCommand(exitCmd);
```

```
  repeat
    clicked := getClickedCommand;
  until clicked <> emptyCommand;
end.
```

```
function getFormTitle: string;
```

عنوان فورم جاری را برمیگرداند

```
function getTextBoxString:string;
```

متن درون تکست باکس را برمیگرداند

```
var cont : command;
    quote : string;
begin
  showTextBox('Enter your favorite quote', 'To be or not to be', 200, TF_ANY);
  cont := createCommand('Continue', CM_SCREEN, 1);
  addCommand(cont);
  repeat
    delay(100);
  until getClickedCommand <> emptyCommand;
  quote := getTextBoxString;
  ...
end.
```

```
function menuAppendString(text: string): integer;
```

پیوست یک ایتم به منو و برگرداندن ایدی ایتم جدید

```
var tetris, minesweeper, snake : integer;
    play, clicked : command;
begin
```

```
  showMenu('Select a game', CH_IMPLICIT);
  tetris := menuAppendString('Tetris');
  minesweeper := menuAppendString('Minesweeper');
  snake := menuAppendString('Snake');
  play := createCommand('Play', CM_SCREEN, 1);
  addCommand(play);
  repeat
    delay(100);
    clicked := getClickedCommand;
  until clicked = play;

  showCanvas; // show canvas and remove menu from the screen
  if menuGetSelectedIndex = tetris then playTetris;
  if menuGetSelectedIndex = minesweeper then playMinesweeper;
  if menuGetSelectedIndex = snake then playSnake;
  ...
end.
```

```
function menuAppendStringImage(text: string; img:image): integer;
```

پیوست کردن یک ایتم با عنوان و عکس به منو و برگرداندن ایدی ایتم جدید

```
var tetris, minesweeper, snake : integer;
    play, clicked : command;
begin
  showMenu('Select a game', CH_IMPLICIT);
  tetris := menuAppendStringImage('Tetris', loadImage('/tetris.png'));
  minesweeper := menuAppendStringImage('Minesweeper', loadImage('/mine.png'));
  snake := menuAppendStringImage('Snake', loadImage('/snake.png'));
  play := createCommand('Play', CM_SCREEN, 1);
  addCommand(play);
  repeat
```

```
    delay(100);
    clicked := getClickedCommand;
  until clicked = play;
  showCanvas; // show canvas and remove menu from the screen
  if menuGetSelectedIndex = tetris then playTetris;
  if menuGetSelectedIndex = minesweeper then playMinesweeper;
  if menuGetSelectedIndex = snake then playSnake;
  ...
end.
```

```
function menuGetSelectedIndex: integer;
```

ایدی منوی انتخاب شده را برمیگرداند اگر منویی انتخاب نشده باشد مقدار 1- را برمیگرداند

```
var tetris, minesweeper, snake : integer;
  play, clicked : command;
begin
  showMenu('Select a game', CH_IMPLICIT);
  tetris := menuAppendStringImage('Tetris', loadImage('/tetris.png'));
  minesweeper := menuAppendStringImage('Minesweeper', loadImage('/mine.png'));
  snake := menuAppendStringImage('Snake', loadImage('/snake.png'));
  play := createCommand('Play', CM_SCREEN, 1);
  addCommand(play);
  repeat
    delay(100);
    clicked := getClickedCommand;
  until clicked = play;
  showCanvas; // show canvas and remove menu from the screen
  if menuGetSelectedIndex = tetris then playTetris;
  if menuGetSelectedIndex = minesweeper then playMinesweeper;
  if menuGetSelectedIndex = snake then playSnake;
  ...
```

```
end.
```

function menuIsSelected(index: integer): boolean;

اگر اندیس مورد نظر در منو انتخاب شده باشد مقدار درستی در غیر این صورت مقدار نادرستی برمیگرداند

```
var tetris, minesweeper, snake : integer;
  play, clicked : command;
begin
  showMenu('Select a game', CH_IMPLICIT);
  tetris := menuAppendStringImage('Tetris', loadImage('/tetris.png'));
  minesweeper := menuAppendStringImage('Minesweeper', loadImage('/mine.png'));
  snake := menuAppendStringImage('Snake', loadImage('/snake.png'));
  play := createCommand('Play', CM_SCREEN, 1);
  addCommand(play);
  repeat
    delay(100);
    clicked := getClickedCommand;
  until clicked = play;
  showCanvas; // show canvas and remove menu from the screen
  if menuIsSelected(tetris) then playTetris;
  if menuIsSelected(minesweeper) then playMinesweeper;
  if menuIsSelected(snake) then playSnake;
  ...
end.
```

procedure playAlertSound;

اجرای یک صدا وابسته به اخطار

```
var cm : command;

begin
  showAlert('Message', 'New message arrived', loadImage('/img1.png'), ALERT_INFO);
  playAlertSound;
```

```
  cm := createCommand('OK', CM_OK, 1);
  addCommand(cm);

  repeat
    delay(100);
  until getClickedCommand <> emptyCommand;

  showForm; // this will clear alert from the screen
  ...
end.
```

```
procedure removeCommand(cmd: command);
```

حذف یک کامند از صفحه نمایش دستگاه

```
procedure removeFormTitle;
```

حذف عنوان فورم

```
procedure setFormTitle(title: string);
```

تغییر عنوان فورم

```
procedure showAlert(title: string; message:string; img:image; alertType:alert);
```

نمایش پیغام بر روی صفحه title عنوان پیغام message متن پیغام alertType میتواند یکی از مقادیر زیر را دارا باشد

ALERT_INFO	اطلاعات
ALERT_WARNING	اخطار
ALERT_ERROR	خطا

ALERT_ALARM پیغام

ALERT_CONFIRMATION تایید

```
var cm : command;

begin
  showAlert('New message', 'You have just received a message from MrSmith',
loadImage('/img1.png'), ALERT_INFO);
  playAlertSound;

  cm := createCommand('Read', CM_OK, 1);
  addCommand(cm);

  repeat
    delay(100);
  until getClickedCommand <> emptyCommand;

  showForm; // this will clear alert from the screen
  ...
end.
```

procedure showCanvas;

ترسیم بر روی صفحه نمایش دستگاه.دستگاه میتواند یک فورم با کلیه اجرا یا یک ترسیم با کلیه اجزا را نمایش دهد

```
var label_id, textField_id: integer;
begin
  label_id := formAddString('Hello world');
  textField_id := formAddTextField('Enter your name', 'Mr.Smith', 20, TF_ANY);
  showForm;
```

```
  delay(2000);
  showCanvas;
  drawText('Hello world', 0, 0);
  repaint;
  delay(2000);
end.
```

```
procedure showForm;
```

از این تابع برای نمایش یک فورم روی صفحه نمایش دستگاه استفاده میشود.یک فورم میتواند دارای المانها و رابطهای کاربری مانند تکست فیلد و دیتا فیلد و ... باشد

```
var label_id, textField_id: integer;
begin
  label_id := formAddString('Hello world');
  textField_id := formAddTextField('Enter your name', 'Mr.Smith', 20, TF_ANY);
  showForm;
  delay(2000);
end.
```

```
procedure showMenu(title:string; menuType:integer);
```

یک منو را در صفحه نمایش دستگاه نشان میدهد.menuType میتواند یکی از مقادیر زیر باشد

CH_IMPLICIT – انتخاب دلخواه

CH_EXCLUSIVE – به هر ایتم یک رادیو باتن اضافه میشود

CH_MULTIPLE – شما میتوانید چند ایتم انتخاب کنید

```
var tetris, minesweeper, snake : integer;
   play, clicked : command;
begin
  showMenu('Select a game', CH_IMPLICIT);
```

```
  tetris := menuAppendString('Tetris');
  minesweeper := menuAppendString('Minesweeper');
  snake := menuAppendString('Snake');

  play := createCommand('Play', CM_SCREEN, 1);
  addCommand(play);

  repeat
    delay(100);
    clicked := getClickedCommand;
  until clicked = play;

  showCanvas; // show canvas and remove menu from the screen

  if menuGetSelectedIndex = tetris then playTetris;
  if menuGetSelectedIndex = minesweeper then playMinesweeper;
  if menuGetSelectedIndex = snake then playSnake;
  ...
end.
```

```
procedure showTextBox(title: string; initialContents: string; maxSize: integer; constraints:
integer);
```

نمایش تکست باکس در صفحه.تکست باکس کل صفحه نمایش دستگاه را اشغال کرده و به جز کامند چیزی روی آن قرار نمیگیرد.مقدار constraints میتواند یکی از مقادیر زیر را دارا باشد

TF_ANY – هر کاراکتری مجاز است

TF_EMAIL – فقط آدرس ایمیل در آن قرار گیرد

TF_NUMERIC – فقط عدد در آن قرار گیرد

TF_PHONENUMBER – فقط شماره تماس در آن قرار گیرد

TF_URL – فقط آدرس وبURL در آن قرار گیرد

مثال:

```
var cont : command;
    quote : string;
begin
  showTextBox('Enter message, '', 200, TF_ANY);
  cont := createCommand('Send', CM_SCREEN, 1);
  addCommand(cont);
  repeat
    delay(100);
  until getClickedCommand <> emptyCommand;

  quote := getTextBoxString;
  ...
end.
```

توابع General

این تابع برای ایجاد تاخیر است که زمان تاخیر برحسب میلی ثانیه بعنوان پارامترورودی تابع محسوب میشود. در مثال زیر یک پیغام به مدت 2ثانیه برروی صفحه نمایش داده میشود و بعد برنامه بسته خواهد شد.یک نکته قابل ذکر : کسانی که با Turbo c و کلا برنامه های کنسول از خانواده c زا کار کرده باشند می دانند که برنامه به طور خودکار بعد اجرا بسته میشود و برای دیدن خروجی برنامه باید از دستوری به نام getch() استفاده کنند.این قضیه هم در اینجا اتفاق می افتد که می بایست عملی مانند getch() را در اینجا پیاده سازی کنیم.

```
procedure delay(millis: integer);

example:
  begin
    drawText('Hello world', 0, 0);
    repaint;
    delay(2000);
  end.
```

این تابع نیز برای گرفتن مشخصاتی از سیستم جاوا گوشی می باشد.که برای اطلاعات بیشتر می توانید از داکیومنت های موجود جاوا استفاده نمایید.

```
function getProperty(propertyName: string): string;
```

برای پایان دادن به کاریک برنامه است.

```
procedure halt;
```

برای انجام دادن بایت کدهای جاوا است.بایت کدهای مورد نظرتان را که می خواهید اجرا شوند باید بعنوان پارامتر به این تابع بفرستید.

```
procedure inline(java_bytecode_statements);
```

مثال:

```
inline(nop;);
```

برای استفاده از بایت کدها میتوانید با استفاده از دستور inline بطور مستقیم بایت کدها را در پروسیجرها بنویسید:

```
procedure DoNothing;
begin
inline(
   nop;
);
end;
```

می توانید از bytecode/end نیز استفاده نمایید:

```
procedure DoNothing;
begin
  bytecode
    nop;
  end;
end;
```

مشخص میکند که آیا MIDlet حالت توقف بوجود آمده است یا نه؟

function isMidletPaused: boolean;

حالت توقف می تواند هنگام داشتن یک تماس ورودی اتفاق بیفتد.که MIDlet به حالت توقف رفته بعد از پاسخ دادن به تماس به حالت resume در خواهد آمد.اگر یک بازی نوشته ایم پس هنگامیکه MIDlet به حالت توقف در آمد بازی مانیز باید متوقف شود.در تکه کد زیر این حالت را پیش بینی کرده و کد آن را نوشته ایم:

example:

```
...
repeat
  { process keypad inputs and read the timer }

  { if the MIDlet is paused, wait until it is resumed }
  while isMidletPaused do
  begin
    delay(100);
  end;

until gameOver;
...
```

تبدیل کد عددی ASCII به مقدار کاراکتر آن

توجه داشته باشید مقدار n نباید از 127 بزرگتر باشد

function chr(n: integer): char;

تبدیل یک کاراکتر به کد عددی یا مقدار ASCII آن

function ord(c: char): integer;

توابع کار با string

این تابع متن بین begin و end را برمیگرداند

function copy(str1: string; begin, end: integer): string;

مثال:

```
Begin
copy('MIDletPascal', 2, 5);
end;
```

مقدار Dle را برمیگرداند

کاراکتر مورد نظر در محل pos را برمیگرداند

function getChar(str: string; pos:integer): char;

مثال:

```
Begin
getChar ('MIDletPascal', 2); //  I
end;
```

تبدیل مقدار عددی به رشته

function integerToString(val: integer): string;

مثال:

```
Var
S:string;
Begin
S:= integerToString (2);
end;
```

طول یک رشته را برمیگرداند

function length(str: string): integer;

مثال:

```
Begin
length ('MIDletPascal'); //  12
end;
```

تبدیل حروف بزرگ به کوچک

function locase(str: string): string;

تبدیل حروف کوچک به بزرگ

function upcase(str: string): string;

جستجوی یک رشته در رشته دیگر مکان اولین جایی که رشته مورد نظر یافت شد را برمیگرداند اگر یافت نشود مقدار 0 برمیگرداند

function pos(str1, str2: string): integer;

مثال:

```
Begin
If pos ('MIDletPascal',’Pa’)=0 then
Halt;
end;
```

اگر مقدار Pa در متن یافت نشود برنامه بسته میشود.توجه داشته باشید که این تابع به حروف کوچک و بزرگ حساس میباشد

کاراکتر ادرس pos را با کاراکتر C در رشته str عوض میکند

function setChar(str: string; c: char; pos:integer): string;

مقدار یک رشته را به عدد صحیح تبدیل میکند

function stringToInteger(s:string):integer;

مثال:

```
Var
i:integer;
Begin
i:= stringToInteger(‘24’);
End;
```

مقدار یک رشته را به یک عدد اعشاری تبدیل میکند

function stringToReal(str: string; base: integer): real;

توابع Debug

ورودی این تابع یک شرط است درست بودن و یا نبودن آن را در پنجره debug ببینید.مثال:

procedure assert(cond: boolean);

example:
Assertion failed at: Tetris.mpsrc:162

این تابع متن مورد نظر را در پنجره debug نمایش میدهد

procedure debug(s: string);

کدهایDrawing

طول صفحه نمایش را به پیکسل برمیگرداند.

function getHeight: integer;

عرض صفحه نمایش را به پیکسل برمیگرداند.

function getWidth: integer;

برای تنظیم رنگ است که قل از اعمال ترسیم می بایست توسط این دستور رنگ موردنظرمان را تنظیم نماییم:
هرکدام از ورودی ها عددی بین 0-255 می گیرد ورودی (255,255,255) رنگ سفید را مشخص مینماید.

procedure setColor(red, green, blue:integer);

این تابع برای رسم یک قوس و کمان است 'startAngle .شروع و arcAngle درجه را مشخص می کند
0درجه در مکان عدد سه ساعت است و 90درجه در مکان عدد 12 ساعت است .مثال:

procedure drawArc(x, y, width, height, startAngle, arcAngle: integer);

example:

```
 begin
  drawArc(0, 0, getWidth, getHeight, 0, 90);
  repaint;
  delay(1000);
 end.
```

برای ترسیم یک بیضی درون صفحه نمایش.مثال:

```
procedure drawEllipse(x, y, width, height: integer);

example:
 begin
  drawEllipse(0, 0, getWidth, getHeight);
  repaint;
  delay(1000);
 end.
```

برای ترسیم یک عکس که از مختصات x,y ترسیم شروع میشود:

```
procedure drawImage(img: image; x, y: integer);

example:
 begin
  drawImage(loadImage('/logo.png'), 0, 0);
  repaint;
  delay(1000);
 end.
```

برای کشیدن یک خط بین دو نقطه('x1', 'y1') و('x2', 'y2').

```
procedure drawLine(x1, y1, x2, y2: integer);

example:
 begin
  drawLine(10, 15, 25, 35);
  repaint;
```

```
  delay(1000);
 end.
```

برای ترسیم خط بیرونی یک مستطیل .مانند:

```
procedure drawRect(x, y, width, height: integer);

example:
 begin
  drawRect(5, 5, 20, 20);
  repaint;
  delay(1000);
 end.
```

ترسیم مستطیل با گوشه های گرد.

```
procedure drawRoundRect(x, y, width, height, arcWidth, arcHeight: integer);

example
 begin
  drawRoundRect(5, 5, 20, 20, 2, 2);
  repaint;
  delay(1000);
 end.
```

متن درون بافر text را برروی صفحه نمایش نشان میدهد

```
procedure drawText(text: string, xPos, yPos: integer);

example:
 begin
  setColor(255, 0, 0);
  drawText('Hello world', 0, 0);
  repaint;
```

```
  delay(1000);
 end.
```

ترسیم بیضی توپر.

```
procedure fillEllipse(x, y, width, height: integer);

 begin
  fillEllipse(0, 0, getWidth, getHeight);
  repaint;
  delay(1000);
 end.
```

ترسیم یک مستطیل توپر.

```
procedure fillRect(x, y, width, height: integer);

example:
 begin
  fillRect(5, 5, 20, 20);
  repaint;
  delay(1000);
 end.
```

ترسیم مستطیل توپر با گوشه های گرد.

```
procedure fillRoundRect(x, y, width, height, arcWidth, arcHeight: integer);

example:
 begin
  fillRoundRect(5, 5, 20, 20, 2, 2);
  repaint;
  delay(1000);
 end.
```

گرفتن هرکدام از رنگهای قرمز و آبی و سبز از رنگ RGB که برای آخرین بار ست شده است.توسط تابع Setcolorرنگ های ست میشوند برای رنگ متن ها،پرکردن اشکال هندسی و...

```
function getColorRed: integer;
function getColorBlue: integer;
function getColorGreen: integer
```

example:

```
 begin
   setColor(127, 0, 0);
   if getColorRed <> 127 then
   begin
     drawText('This should never happen', 0, 0);
     repaint;
   end;
   delay(1000);
 end.
```

طول یک عکس به پیکسل برمیگرداند.

```
function getImageHeight(img: image): integer;
```

عرض یک عکس را به پیکسل برمیگرداند.

```
function getImageWidth(img: image): integer;
```

طول و عرض متن نمایش داده شده به فونت جاری را به پیکسل بر میگرداند.

```
function getStringHeight(text: string): integer;
function getStringWidth(text: string): integer;
```

example:

```
var text: string;
  height: integer;
begin
  text   := 'Text to display';
  height := getStringHeight(text);
  drawText(text, 0, (getHeight - height)/2);
  repaint;
  delay(1000);
end.
```

ایجاد یک عکس با مختصات تعیین شده از بوم رنگ شده تان.

```
function ImageFromCanvas(x: integer; y: integer; width: integer; height: integer ): image;
```

ایجاد یک عکس با مختصات تعیین شده از عکسی دیگر که برای برش عکس ها جالب است.

```
function ImageFromImage(sourceImg: image; x: integer; y: integer; width: integer; height:
integer ): image;
```

اگر دستگاه قادر به نمایش رنگ موردنظرمان باشد مقدار true بر میگرداند.

```
function isColorDisplay:boolean;
```

تعداد رنگهایی که دستگاه قادر به نمایش انها میباشد

```
function getColorsNum:integer;
```

برای نمایش عکس از ریسورس.مثال:

```
function loadImage(resource: string): image

example:
  begin
    drawImage(loadImage('/icon.png'), 0, 0);
    repaint;
```

```
  delay(1000); { wait 1 second before the MIDlet terminates }
 end.
```

ایجاد یک محدوده یا قاب برای رسم

```
procedure setClip(int x, int y, int width, int height);
```

این تابع یک محدوده ایجاد میکند و تمتم اشکال و ترسیمها فقط درون این محدوده انجام میشود و خارج از این محدوده تغییری نمیکند. مثال:

```
 begin
  // draw the ellipse
  setColor(0, 0, 0);
  fillEllipse(0, 0, getWidth, getHeight);
  repaint;
  delay(2000);

  // draw ellipse again with setClip called before
  setClip(10, 10, 15, 25);
  fillEllipse(0, 0, getWidth, getHeight);
  repaint;
  delay(2000);
 end.
```

تغییر رنگ یک پیکسل از صفحه نمایش:

```
procedure plot(x, y:integer);

example:
 begin
  setColor(255, 0, 0);
  plot(5, 10);
  repaint;
  delay(1000);
 end.
```

تمام دستورات ترسیم مانند drawLine, drawText, fillRect و ... بطور مستقیم برروی صفحه نمایش اعمال نمیشوند که off-screen buffer برروی صفحه نمایش نشان داده میشوند.این دستور بطوری صفحه نمایش را بروز کرده تا تغییرات اعمال شده را نمایش دهد.

procedure repaint;

اعمال فونت پیش فرض که اگر کاربر فونتی را تعیین کرده باشد به حالت پیش فرض برخواهد گشت.

procedure setDefaultFont;

برای تعیین فونت صفحه بکاربرده میشود.

procedure setFont(fontFace, fontStyle, fontSize);

پارامتر fontFace میتواند مقادیر زیر را بگیرد:

- FONT_FACE_SYSTEM
- FONT_FACE_MONOSPACE
- FONT_FACE_PROPORTIONAL

پارامتر fontStyle هم مقادیرزیر را میتواند بگیر:

- FONT_STYLE_PLAIN
- FONT_STYLE_BOLD
- FONT_STYLE_ITALIC
- FONT_STYLE_UNDERLINED

برای استفاده چندتای این گزینه بدین صورت عمل میکنیم.مثلا میخواهیم فونت مان ضخیم و زیر خط دار باشد

: (FONT_STYLE_BOLD or FONT_STYLE_UNDERLINE)

و پارامتر fontSize نیز دارای مقدارهای زیر می باشد:

- FONT_SIZE_SMALL
- FONT_SIZE_MEDIUM
- FONT_SIZE_LARGE

مثال:

```
begin
   setFont(FONT_FACE_SYSTEM, FONT_STYLE_BOLD or FONT_STYLE_UNDERLINED,
FONT_SIZE_LARGE);
   drawText('Hello world', 0, 0);
   repaint;
   delay(1000);
 end.
```

صفحه کلید و کلیدها

function getKeyClicked: integer;

مقدار کد اخرین کلید فشرده شده را برمیگرداند

function getKeyPressed: integer;

مقدار کد کلیدی که هم اکنون فشرده شده را برمیگرداند .اگر کاربر کلیدی را فشار دهد مقدار آن را برمیگرداند که KE_NONE یعنی هیچ کلیدی فشرده نشده است.مقدارهای پیش فرض برای صفحه کلید گوشی:

- KE_KEY0
- KE_KEY1
- KE_KEY2
- KE_KEY3
- KE_KEY4
- KE_KEY5

• KE_KEY6
• KE_KEY7
• KE_KEY8
• KE_KEY9
• KE_STAR
• KE_POUND

درمثال زیر تا کلید * را فشارندهید از برنامه خارج نخواهد شد که همان شبیه GETCH() عمل میکند:

```
begin
   while getKeyClicked <> KE_STAR do
   begin
     delay(100);
   end;
 end.
```

```
function keyToAction(keyCode: integer): integer;
```

گاهی کلیدهایی هستند که برای گوشی های مختلف مقدارهای گوناگون برمیگرداند مثلا کلید fire ممکن است برای یک گوشی مقدار 10 و برای گوشی دیگری مقدار 120را برگرداند.که برایاین مشکل از تابع ذکرشده استفاده میکنیم:

• GA_NONE
• GA_UP
• GA_DOWN
• GA_LEFT
• GA_RIGHT
• GA_FIRE
• GA_GAMEA

• GA_GAMEB
• GA_GAMEC
• GA_GAMED

در مثال زیر وقتی کلید fire زده شود برنامه بسته خواهد شد:

```
begin
  while keyToAction(getKeyClicked) <> GA_FIRE do
    begin
      delay(100);
    end;
  end.
```

دستورات کار با زمان و تاریخ

function getCurrentTime: integer;

این تابع زمان را از نیمه شب 1.1.1970 تا به حال را برمیگرداند که توسط توابع در ادامه توضیح داده خواهد شد می توانید ساعت و دقیقه و ... را ازدرون این مقدار برگشتی بیرون بکشید.

function getSecond(time: integer): integer;

مقدار ثانیه را از time را برمیگرداند که عددی بین 0 تا 59 میباشد

function getHour(time: integer): integer;

مقدار ساعت را از یک متغیر time برمیگرداند که عددی بین 0 تا23 است

function getMinute(time: integer): integer;

عددی را بر میگرداند که دقیقه است و عددی بین 0 تا 59 است.

```
var time: integer;
    text: string;
 begin
   time := getCurrentTime;
   text := 'Current time is ' + getHour(time);
   text := text + ':' + getMinute(time);
   text := text + ':' + getSecond(time);
   drawText(text, 0, 0);
   repaint;
   delay(1000); { wait 1 second before MIDlet terminates }
 end.
```

```
function getDay(time: integer): integer;
```

برای برگرداندن روز در ماه جاری از درون تاریخ بدست آمده از تابع. getCurrentTime

```
function getMonth(time: integer): integer;
```

برگرداندن ماه جاری

```
function getWeekDay(time: integer): integer;
```

برگرداندن روز هفته که عدد 1برای Sunday و عدد 2 برای Monday و تا 7 که Saturday میشود.

```
function getYear(time: integer): integer;
```

برگرداندن سال

function getYearDay(time: integer): integer;

مقدار خروجی تابع تعداد روز سال جاری است که عددی بین 1 تا 366 می باشد.

function getRelativeTimeMs: integer;

زمان جاری را بر اساس میلی ثانیه برمیگرداند.که عددی 32بیتی است و یعنی زمانی 2^32 می تواند برگرداند که میشه 48 روز و هر 48 روز ریست شده و از 0 شروع به شمارش میکند.پس برای تاریخ جاری خوب نیست و میتونید برای پیاده سازی یک تایمر در برنامه استفاده کنید.بعنوان مثال یک بازی ساده مانند tetrisرا درنظر بگیرید که در ثانیه نیاز دارید بلاک ها را حرکت دهید و می توانید توسط کلیدهای فشرده شده بلاک ها را به سمت چپ یا راست حرکت دهید.که حلقه اصلی می تواند بدین صورت باشد:

```
...
 lastSavedTime := getRelativeTimeMs; { initialize the timer }
 repeat
   { read and process the keypad input }
   key := getKeyClicked;
   if keyToAction(key) = GA_LEFT then moveLeft;
   if keyToAction(key) = GA_RIGHT then moveRight;

   { check if 1 second has passed }
   if ((getRelativeTimeMs - lastSavedTime) > 1000)
   or (getRelativeTimeMs < lastSavedTime)  { check if the timer is reset after 48 days }
     then
     begin
       lastSavedTime := getRelativeTimeMs;
       moveDown;
     end;
 until gameOver;
 ...
```

دستورات ریاضی:

function abs(n: integer): integer;

قدرمطلق یک عدد را برمیگرداند

function acos(num: real): real;

آرککوسینوس یک عدد را بر میگرداند که بین 0 تا pi بر حسب رادیان است.

function asin(num: real): real;

function atan(num: real): real;

برگرداندن ارکسینوس و ار کتانژانت در محدوده pi/2- تا pi/2 برحسب رادیان.

function atan2(y, x: real): real;

تبدیل مختصات (x، y) مختصات قطبی (r، تتا). این روش محاسبه فاز تتا با محاسبه مماس بر قوس Y / X در محدوده-PI.

function cos(num: real): real;

برگرداندن کوسینوس یک عدد

function sin(num: real): real;

سینوس عدد گرفته شده را برمیگرداند.

function tan(num: real): real;

تانژانت یک عدد رابرمیگرداند.

function exp(num: real): real;

مقادار تابع نمایی یک عدد را برمیگرداند

function frac(num: real): real;

عددی اعشاری ورودی گرفته و مقدار اعشاری آن را برمیگرداند.

function log(num: real): real;

لگاریتم طبیعی عدد داده شده را برمیگرداند.

function log10(num: real): real;

لگاریتم عدد داده شده در مبنای 10 را برمیگرداند.

function pow(a, b: real): real;

عدد a را به توان b میرساند.

function odd(n: integer): boolean;

اگر مقدار a فرد باشد مقدار true برمیگرداند

function rabs(num: real): real;

قدرمطلق یک عدد حقیقی را برمیگرداند.

function random(n: integer): integer;

عددی تصادفی بین تا n-1 برمیگرداند.

procedure randomize;

سازنده اعدا تصادفی را ازنوع مقداردهی میکند

function sqr(n: integer): integer;

مربع عدد داده شده را برمیگرداند.

function sqrt(num: real): real;

ریشه دوم عدد را برمیگرداند.

function toDegrees(num: real): real;

تبدیل زاویه از رادیان به درجه.

function toRadians(num: real): real;

تبدیل زاویه از درجه به رادیان

function trunc(num: real):integer;

قسمت صحیح یک عدد حقیقی را برمیگرداند.

توابع ارسال sms

ارسال یک sms اگر sms با موفقیت به سرور ارسال شود مقدار درستی در غیر این صورت مقدار نادرستی را برمیگرداند

function smsStartSend(destination: string; message:string): boolean;

مثال:

```
begin
if not smsStartSend('sms://+5550000', 'Hello!') then halt;
  while smsIsSending do // wait for the message to be sent
    delay(100);
    if not smsWasSuccessfull then halt; // check if the message was sent successfully
end.
```

چک کردن اینکه ایا در حال ارسال sms هستید

function smsIsSending: boolean;

چک کردن اینکه ایا sms با موفقیت ارسال شده یا نه

function smsWasSuccessfull: boolean;

ارسال نشدن ممکن است به خاطر پشتیبانی نکردن از ارسال با J2ME MIDlet باشد

- ادرس ارسالی صحیح نمیباشد
- اپراتور در ارسال با مشکل مواجه شود

توابع کار با sound

گرفتن طول یک اهنگ که در پلیر لود شده بر حسب میلی ثانیه

function getPlayerDuration:integer;

باز کردن یک ریسورس در پلیر موبایل

function openPlayer(resource:string; mimetype:string):boolean;

مقدار mimetype میتواند به صورت زیر باشد

wave files: audio/x-wav

au files: audio/basic

MP3 files: audio/mpeg

MIDI files: audio/midi

مثال:

```
begin
  if not openPlayer('/explosion.mid', 'audio/midi') then  halt;
  if not setPlayerCount(-1) then halt;
  if not startPlayer then halt;
  delay(5000);
end.
```

این تابع همانند دکمه تکرار عمل میکند و میتوانید مشخص کنید یک اهنگ چند بار تکرار شود

```
function setPlayerCount(loopCount:integer):boolean;
```

اجرای آهنگ لود شده در پلیر

```
function startPlayer:boolean;
```

اجرای آهنگ در پلیر را متوقف میکند

```
procedure stopPlayer;
```

نوشتن اولین برنامه

نرم افزار MIDletPascal را باز کنید. از منوی فایل new و سپس project را بزنید. و نامی برای پروژه خود انتخاب کنید. و okکنید حالا در محیط کد نویسی کدهای زیر را بنویسید و برای اجرا از منوی Project ابتدا Build Projectرا بزنید و سپس Run MIDlet را بزنید. یا می توانید Build and Run را بزنید. که هر دو کار را با هم انجام بدهید. یا با کلید F9 همین کار را انجام خواهید داد یا از ساید باز سمت راست یکی از این

دکمه ها را انتخاب کنید

```
program HelloWorld;

begin

  DrawText ('Hello, World!', 0, 0);

  Repaint;

  Delay(2000);

end.
```

بعد از اجرا فایل نهایی ساخته شده و ادرس ان در انتهای پنجره Message نمایش داده میشود مانند زیر

Launching cmd /A /C "C:\NewProject\bin\HelloWord.jad"

که فایل مورد نظر را از ادرس فوق برداشته و به گوشی منتقل کنید

کشیدن دایره:

```
program NewProject;

begin

   drawEllipse(0, 0, getWidth, getHeight);

   repaint;

   delay(6000);

end.
```

یک دایره می کشد.

نوشتن function و procedure

Proceduresها زیر برنامه هایی هستند که مقدار خروجی ندارند وبطورکلی چیزی را برنمی گردانند.درمثال زیر دو پروسیجر نوشته شده است است که یکی از آنها بدون پارامتر ورودی و دیگری دارای دوپارامتر ورودی است:

```
program procedureSample;
 var
n: integer; { this variable is visible in main
              program block and in all functions
              and procedures }

 procedure noArgs;
 begin
  n := 5;
 end;

 procedure twoArgs(a: integer; b: string);
 var len: integer; { this variable can be accessed only
                  from within the procedure and is
                  reinitialized every time that the
                  procedure is called }
 begin
  len := length(b);
  n := a + len;
 end;

 begin
  noArgs; { call the first procedure }
  twoArgs(n, 'Some string'); { call the second procedure }
 end.
```

و Functions ها زیر برنامه هایی هستند که مقداری را بعنوان خروجی باید برگردانند Functions .همانند پروسیجرها هستند ولی با این تفاوت که مقداری را باید بعنوان خروجی برگردانند.

```
program functionSample;
 var result: integer;

 { this simple function always returns 5 }
 function returnFive: integer;
 begin
  returnFive := 5;
 end;

 function multiply(a, b: integer): integer;
 begin
  multiply := a * b;
 end;

 begin
  result := multiply(2, returnFive); { call the multiply function
                                       where on argument is value
                                       returned by calling the
                                       other function }
 end.
```

می توانید از کلمه کلیدی result برای برگرداندن مقدار در توابع استفاده نمایید.در مثال زیر طرز استفاده را خواهید دید:

```
function Return4: integer;
begin
 result := 4;
end;
```

برای خروج از یک تابع را رویه در هر لحظه میتوانید از دستور Exit استفاده کنید

```
Procedure Exit;
```

همچنین می توان func هایی بصورت بازگشتی نوشت.همانند زیر که حالت فاکتوریل را پیاده سازی کرده ایم:

```
program recursionSample;
 var factorielOfFive: integer;
 function factoriel(n: integer): integer;
 begin
  if n = 1 then
    factoriel := 1;
  else
    factoriel := n * factoriel(n-1);
 end;

 begin
  factorielOfFive := factoriel(5);
 end.
```

حالت زیر را درنظر بگیرید:

```
procedure a (x: integer);
 begin
  ...
  b(x);
  ...
 end;

 procedure b(y: integer);
 begin
 ...
  a(y);
 ...
```

```
end;

begin
  a(5);
end.
```

خوب همانطورکه میبینید پروسیجر a پروسیجر b را فراخوانی میکند و باز از درون پروسیجر b نیز پروسیجر a نیز فراخوانی میشود.که کامپایلر پیغام خطا میدهد که برای رفع این مشکل بصورت زیر عمل می کنیم:

```
procedure b(y:integer); forward;{ the forward reference tells
                                  compiler that procedure 'b' will
                                  be defined somewhere in the program code }

procedure a (x: integer);
begin
  ...
  b(x);
  ...
end;

procedure b(y: integer);
begin
  ...
  a(y);
  ...
end;

begin
  a(5);
end.
```

که این تکنیک و حالت را forward references گویند.

چند سورس ساده

سورس برنامه ماشین حساب

```
program Calculator;
  var textField_1,textField_3,choice_1,
      textField_2,sum,sub,mul,diw: Integer;
      Result:Real;
      exitCmd, okCmd, clicked: command;
begin
showForm;
  SetTicker('MB ماشین حساب');
  textField_1:=FormAddTextField('20,'',': ورودی عدد اول,TF_NUMERIC);
  textField_2:=FormAddTextField('20,'',': ورودی عدد دوم,TF_NUMERIC);
  choice_1:=FormAddChoice('انتخاب عملیات:',CH_EXCLUSIVE);
   sum := ChoiceAppendStringImage(choice_1, 'جمع(+)',LoadImage('/icon.png'));
   sub := ChoiceAppendStringImage(choice_1, 'تفریق(-)',LoadImage('/icon.png'));
   mul := ChoiceAppendStringImage(choice_1, 'ضرب(*)',LoadImage('/icon.png'));
   diw := ChoiceAppendStringImage(choice_1, 'تقسیم(/)',LoadImage('/icon.png'));
```

```
  textField_3:=FormAddTextField('20,'',':نتیجه,TF_ANY);

  exitCmd:=CreateCommand('خروج', CM_EXIT, 1);

  AddCommand(exitCmd);

  okCmd:=CreateCommand('محاسبه', CM_OK, 1);

  AddCommand(okCmd);

   repeat

     clicked := GetClickedCommand;

     if clicked = okCmd then

     Begin

     if choiceIsSelected(choice_1, sum) then

   FormSetText(textField_3, IntegerToString(StringToInteger(FormGetText(textField_1)) +
StringToInteger(FormGetText(textField_2))));

   else

     if choiceIsSelected(choice_1, sub) then

   FormSetText(textField_3, IntegerToString(StringToInteger(FormGetText(textField_1)) -
StringToInteger(FormGetText(textField_2))));

   else

     if choiceIsSelected(choice_1, mul) then

   FormSetText(textField_3, IntegerToString(StringToInteger(FormGetText(textField_1)) *
StringToInteger(FormGetText(textField_2))));

   else

     if choiceIsSelected(choice_1, diw) then

     Begin
```

```
    if FormGetText(textField_2)='0' then
   FormSetText(textField_3, 'تعریف نشده');
       else
   FormSetText(textField_3, IntegerToString(StringToInteger(FormGetText(textField_1)) /
StringToInteger(FormGetText(textField_2))));
 End;
    End;
   until clicked = exitCmd;
end.
```

Screen Saverساده

کد زیر با استفاده از یک حلقه ,متنی را تا زمانی که کلیدی فشرده نشود نشان می دهد

```
program Screen_Saver;
begin
  while GetKeyPressed = KE_NONE do
    begin
      DrawText('Hello MidletPascal ', 70, 130);
      Repaint;
      Delay(100);
    end;
end.
```

خواندن فایل متنی و نمایش 2 خط اول آن روی صفحه

```
program ResourceSample;
var text: Resource;
 img: image;
 exit: command;
 str1, str2: string;
begin
 text: = OpenResource ('/ tekst.txt');
 if (ResourceAvailable (text)) then
  begin
   str1: = ReadLine (text);
   str2: = ReadLine (text);
   CloseResource (text);
  end;
 DrawText (str1, 0,130);
 DrawText (str2, 0,140);
 img: = LoadImage ('/ image.png');
 DrawImage (img, 0,0);
 if OpenPlayer ('/ wrong.mid', 'audio / midi') then
  if SetPlayerCount (1) then
   if StartPlayer then DrawText ('duration -' +
    IntegerToString (GetPlayerDuration) + 'ms.', 0.150);
```

```
 exit: = createCommand ('output', CM_SCREEN, 1);
 addCommand (exit);
 Repaint;
 while exit <> GetClickedCommand do delay (100);
end.
```

حرکت کادر به عقب و جلو با استفاده از دکمه های کنترلی

```
 program Sample;
 var x, y, key: integer;
begin
 x: = 0;
 y: = 0;
 while key <> GA_FIRE do
  begin
   SetColor (255,255,255);
   FillRect (0,0,240,320);
   SetColor (0,0,0);
   FillRect (x, y, 10,10);
   Repaint;
   Delay (100);
   Key: = KeyToAction (GetKeyClicked);
   if key = GA_LEFT then x: = x-10;
   if key = GA_RIGHT then x: = x +10;
```

```
  if key = GA_UP then y: = y-10;
  if key = GA_DOWN then y: = y +10;
 end;
end.
```

کتابخانه ها

کتابخانه های زیادی هم برای این کامپایلر ایجاد شده و قابلیتهای زیادی روی موبایل به شما میدهد که میتوانید از لینک زیر انها را دریافت کنید البته سایت زیر یک فروم برای تلفنهای همراه میباشد و بخش MidletPascal آن بسیار فعال میباشد

بخش برنامه های نوشته شده

www.boolean.name > Game Programming Mobile > MidletPascal > Our Open Source Software

www.boolean.name > Game Programming Mobile > MidletPascal > Projects MidletPascal

لیست کتابخانه ها

www.boolean.name > Game Programming Mobile > MidletPascal > Libraries

> Working with the file system:

Lib_bmp: Save BMP images into PS

Lib_imload: Download images of FS

Lib_jsr75: Create, read, write files to the file system

Lib_jsr75ex: Extended to work with FS

Lib_jsr75mf: Simultaneous work with several files

Lib_pim: Using the phone book

Lib_png: Save images in PNG format

Lib_jpeg: Save images in JPEG format

> Backup / decompress data

Lib_gzip: Unpacking GZIP

Lib_zip: Unpack the ZIP (jsr75)

Lib_tar: unpacking tar archives

> Network and Bluetooth:

Lib_bt: Full support for bluetooth connection

Lib_netutils: Download images for HTTP

Lib_socket: socket support

Lib_web: Working with http, correct handling of POST

Lib_ftp: Implementing support for FTP protocol

Lib_binsock: send / receive binary data over a socket

Lib_proweb - full support for http. Work POST.

Lib_smtp2 - Sending mail with authorization

> Graphics:

LIb_st: Output styled text

Lib_font32: Graphic fonts

Lib_text: Text output in the transfer window and styles

Lib_alpha: Change the transparency of the image

Lib_gfx: Advanced graphics features

Lib_canvas: Rotate the screen and copy

All you need to work with graphic fonts

Lib_game: Sprites, tiles, etc. Game designer, in general

Lib_anigif: Download and playback is GIF-animation

Lib_jpeg: Saving JPEG-images

Lib_effects: Library to apply different effects to the picture

Lib_m3g: support for 3D graphics

Lib_mcv3: Mascot Capsule v3

Lib_picker: copy of the image with transparency

Lib_gif: create animated GIF images

Lib_beta: Redirects output to the image / display

Lib_rc: screen rotation

Lib_turn: Rotating images at any angle

> Media:

Lib_mmapi: Advanced player controls

Lib_player: Any number of players. Need support Audio Mixing

Lib_tonesnd: tone generation

Lib_media: Audio Capture, photos /? \

Lib_videocnv: Playing video in Canvas

Lib_medialist: Get a list of media formats supported by the phone

> System features:

Lib_thread: Implementation of the procedure in an independent thread

Lib_memory: The number of used and available memory

Lib_memclean: Clearing memory of "rubbish"

Lib_str: Additional functions for string manipulation

Lib_vibra: Vibration Control

Lib_autorun: Autostart MIDlet

Lib_timer: Performing procedures on the timer

Lib_array2d: Creating a two-dimensional dynamic arrays

Lib_bytes: dynamic array of bytes

Lib_dbl: java double support

Lib_rms: New aspects of a record store

Lib_call - A phone call to the number

Lib_scodes - Definition of codes of soft keys and phone model

Lib_resloader - Economical resource utilization

Lib_sms: Sending SMS

Lib_radix - Number Systems

Lib_keys: Pressing the buttons

Lib_suspend: Collapsing MIDlet

Lib_safeload: safe loading of images

Lib_threads - dynamic flow (performing procedures on separate threads, parallel to the main program)

Lib_adata - Several dynamic arrays

Lib_xml - Splitting XML / HTML document

Lib_parse: splitting strings

> Cryptography / coding data

Lib_base64: Implementation of the algorithm base64

Lib_md5hash: implementation of a cryptographic algorithm MD5

Lib_vault - resources under lock and key

> Standard interface:

Lib_ui: Advanced control of the shape and the menu

> Advanced graphical interface:

Lib_gui32 - long suffering, windows

Lib_font32: Graphic fonts

Lib_line32 - Horizontal bar elements

Lib_menu32 - Graphical menus without headaches

پایان