

# WP04 - SENSORI E HARDWARE CON WINDOWS PHONE 8.1



Dan Ardelean

[dan.ardelean@live.com](mailto:dan.ardelean@live.com)

@danardelean

<http://sviluppomobile.blogspot.com>



# Grazie a



## Sponsor



# Agenda

- Bluetooth Low Energy / Smart
- Lumia SensorCore

# Bluetooth Low Energy

- **2001** Nokia inizia lavorare su una nuova tecnologia wireless
- **2004** risultati pubblicati con il nome Bluetooth Low End Extension
- **2006** rilasciato pubblicamente come **Wibree**
- **2007** accordo per includere Wibree dentro **Bluetooth come Ultra low power technology**
- **2010** integrato come **Bluetooth Smart** dentro Core Specifications 4.0
- **2011** viene rilasciato iPhone 4S con supporto per BLE
- **2012** escono i primi dispositivi
- **2013** Windows 8.1 aggiunge supporto parziale per BLE
- **2014** Windows Phone 8.1 aggiunge supporto per BLE

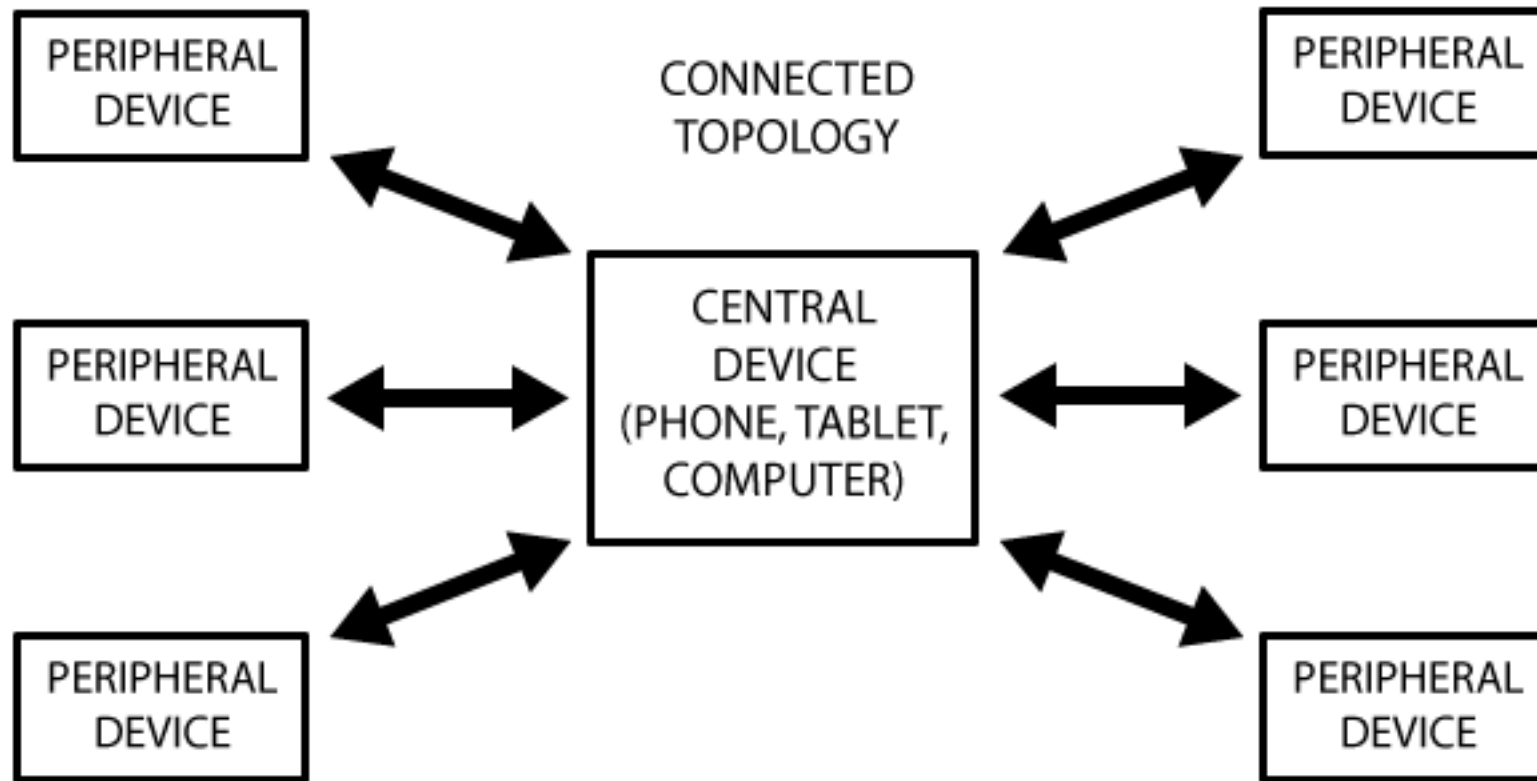
# Bluetooth Low Energy

- “Internet of Things”: low cost, low power
- Proximity and out of range detection (geofencing)
- Fast connections (10 msec) and low data latency (3-6 msec)
- Adaptive Frequency Hopping (AFH), resilient to interference
- Broadcast support (iBeacons)
- Connectionless always off technology
- Low power consumption:  $\sim 30\mu\text{A}$  average consumption 1 second data interval, equivalent to 330 days on a single coin cell
- Range up to 280 meters (in practice about 10-30 meters)

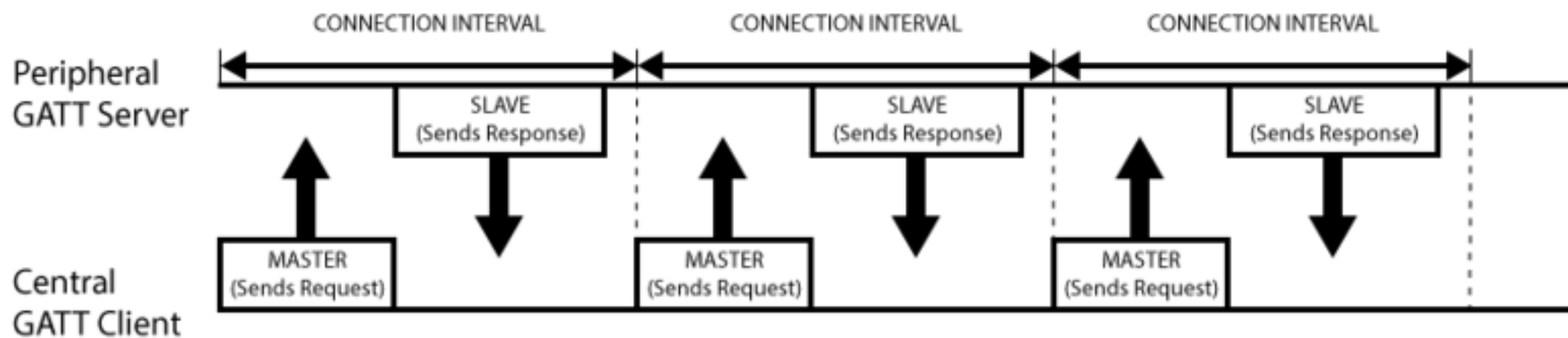
# Classic vs Smart

Technical Specification	Classic Bluetooth technology	Bluetooth Smart technology
Distance/Range (theoretical max.)	100 m (330 ft)	> 100 m (> 330 ft)
Over the air data rate	1–3 Mbit/s	1 Mbit/s
Application throughput	0.7–2.1 Mbit/s	0.27 Mbit/s
Active slaves	7	Not defined; implementation dependent
Security	56/128-bit and application layer user defined	128-bit <a href="#">AES</a> with <a href="#">Counter Mode CBC-MAC</a> and application layer user defined
Robustness	Adaptive fast frequency hopping, <a href="#">FEC</a> , fast <a href="#">ACK</a>	Adaptive frequency hopping, Lazy Acknowledgement, 24-bit CRC, 32-bit Message Integrity Check
Latency (from a non-connected state)	<b>Typically 100 ms</b>	<b>6 ms</b>
Total time to send data (det.battery life)	<b>100 ms</b>	<b>3 ms</b>

# Network Topology

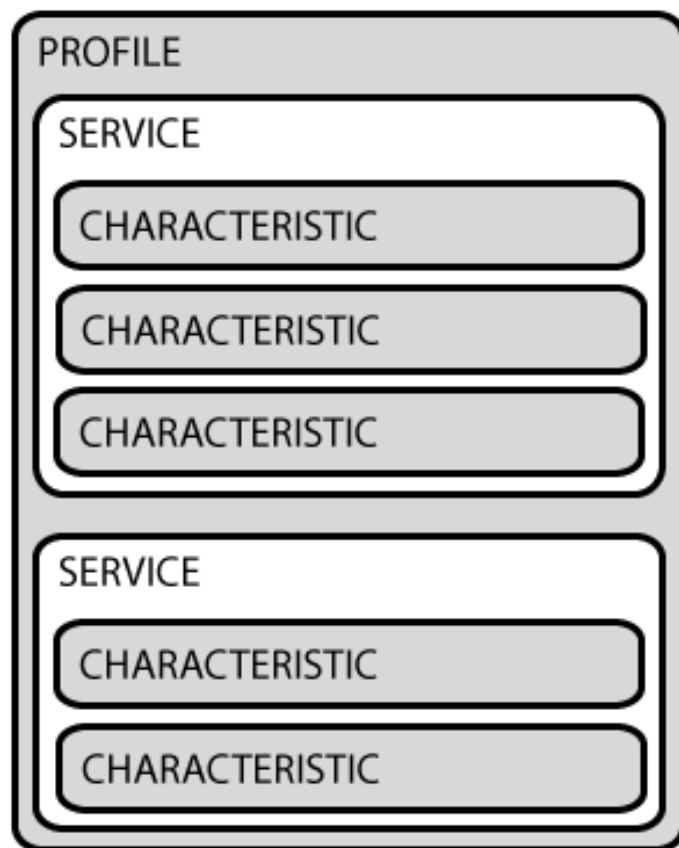


# GATT Transactions





# Profili e caratteristiche



- Profili
  - Collezione predefinita di servizi
  - Non esistono fisicamente
  - HRP = Heart Rate Service + Device Info
- Servizi
  - Sono le entità logiche (gruppi di data)
  - Identificatore univoco UUID
- Caratteristiche
  - I dati del dispositivo
  - Identificatore univoco UUID

# Healthcare

- HTP - for medical temperature measurement devices.
- GLP - for blood glucose monitors.
- BLP - for blood pressure measurement.



iTherm  
MONITORING TEMPERATURE FOR KIDS



# Sport and fitness

- HRP - for devices which measure heart rate.
- CSCP - for sensors attached to a bicycle or exercise bike to measure cadence and wheel speed.
- RSCP - running speed and cadence profile.
- CPP - cycling power profile.
- LNP — location and navigation profile.



# Proximity Sensor

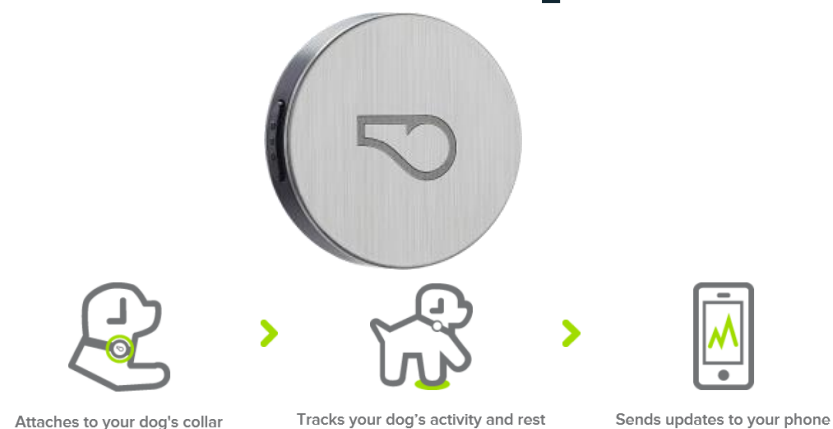
- FMP - the "find me" profile — allows one device to issue an alert on a second misplaced device
- PXP - the proximity profile — allows a proximity monitor to detect whether a proximity reporter is within a close range.



# Home Automation



# Altri dispositivi





# Profil standard

- <https://developer.bluetooth.org/TechnologyOverview/Pages/Profiles.aspx#GATT>

ANP	Alert Notification Profile	HRP	Heart Rate Profile	PXP	Proximity Profile
ANS	Alert Notification Service	HRS	Heart Rate Service	RTUS	Reference Time Update Service
BAS	Battery Service	HIDS	HID Service	ScPP	Scan Parameters Profile
BLP	Blood Pressure Profile	HOGP	HID Over GATT Profile	ScPS	Scan Parameters Service
BLS	Blood Pressure Service	IAS	Immediate Alert Service	TIP	Time Profile
CTS	Current Time Service	LLS	Link Loss Service	TPS	Tx Power Service
DIS	Device Information Service	NDCS	Next DST Change Service		
FMP	Find Me Profile	PASP	Phone Alert Status Profile		
HTP	Health Thermometer Profile	PASS	Phone Alert Status Service		

# Advertiser/Broadcaster

P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData								CRC	RSSI (dBm)	FCS
					Type	TxAdd	RxAdd	PDU-Length												
1	+0 =0	0x27	0x8E89BED6	ADV_IND	0	0	0	24	0x9059AF08A96F	02 01 06 0E FF 0D 00 05 00 05 01 02 35 07 00 00 06 62								0x7BA9E6	-46	OK
P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData								CRC	RSSI (dBm)	FCS
					Type	TxAdd	RxAdd	PDU-Length												
2	+1029365 =1029365	0x27	0x8E89BED6	ADV_IND	0	0	0	24	0x9059AF08A96F	02 01 06 0E FF 0D 00 05 00 05 01 02 35 07 00 00 06 62								0x7BA9E6	-46	OK
P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				Adv PDU Payload	CRC	RSSI (dBm)	FCE								
					Type	TxAdd	RxAdd	PDU-Length												
3	+1027209 =2056574	0x27	0x8E89BED6	Unknown	15	0	1	7	3B DB 9E 6A 68 D6 DB	0xA3DC4E	-85	ERROR								
P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData								CRC	RSSI (dBm)	FCS
					Type	TxAdd	RxAdd	PDU-Length												
4	+3283375 =5339949	0x27	0x8E89BED6	ADV_IND	0	0	0	24	0x9059AF08A96F	02 01 06 0E FF 0D 00 05 00 05 01 02 35 07 00 00 06 62								0x7BA9E6	-46	OK
P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData								CRC	RSSI (dBm)	FCS
					Type	TxAdd	RxAdd	PDU-Length												
5	+1031865 =6371814	0x27	0x8E89BED6	ADV_IND	0	0	0	24	0x9059AF08A96F	02 01 06 0E FF 0D 00 05 00 05 01 02 35 07 00 00 06 62								0x7BA9E6	-46	OK
P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData								CRC	RSSI (dBm)	FCS
					Type	TxAdd	RxAdd	PDU-Length												
6	+1023115 =7394929	0x27	0x8E89BED6	ADV_IND	0	0	0	24	0x9059AF08A96F	02 01 06 0E FF 0D 00 05 00 05 01 02 35 07 00 00 06 62								0x7BA9E6	-46	OK
P.nbr.	Time (us)	Channel	Access Address	Adv PDU Type	Adv PDU Header				AdvA	AdvData								CRC	RSSI (dBm)	FCS
					Type	TxAdd	RxAdd	PDU-Length												
7	+1026240 =8421169	0x27	0x8E89BED6	ADV_IND	0	0	0	24	0x9059AF08A96F	02 01 06 0E FF 0D 00 05 00 05 01 02 35 07 00 00 06 62								0x7BA9E6	-46	OK



# demo

TI Packet Sniffer



# Windows Phone 8.1

- 1. Package.appxmanifest

```
<Capabilities>
  <Capability Name="internetClientServer" />
  <m2:DeviceCapability Name="bluetooth.genericAttributeProfile">
    <m2:Device Id="any">
      <m2:Function Type="name:heartRate"/>
    </m2:Device>
  </m2:DeviceCapability>
</Capabilities>
```

- 2. Trovare dei dispositivi che implementano il servizio

```
var devices = await Windows.Devices.Enumeration.DeviceInformation.FindAllAsync(GattDeviceService.GetDeviceSelectorFromUuid(GattServiceUids.HeartRate));
```

- 3. Realizzare la connessione al servizio desiderato

```
service = await GattDeviceService.FromIdAsync(devices[0].Id);
```

# Windows Phone 8.1

- 4. "Collegamento" alla caratteristica desiderata

```
var cHeart = service.GetCharacteristics(GattCharacteristicUuids.HeartRateMeasurement)[0];  
var bodySensorLocationCharacteristics = service.GetCharacteristics(GattCharacteristicUuids.BodySensorLocation);
```

- 5. Impostare eventualmente il tipo di lettura

```
var currentDescriptorValue = await cHeart.ReadClientCharacteristicConfigurationDescriptorAsync();  
GattCommunicationStatus status = await cHeart.WriteClientCharacteristicConfigurationDescriptorAsync(GattClientCharacteristicConfigurationDescriptorValue.Notify);
```

- 6. Leggere/scrivere o sottoscrivere le notifiche

```
cHeart.ValueChanged += cHeart_ValueChanged;  
GattReadResult readResult = await bodySensorLocationCharacteristics[0].ReadValueAsync();
```

- 7. Interpretare il risultato della lettura

```
var data = new byte[args.CharacteristicValue.Length];  
DataReader.FromBuffer(args.CharacteristicValue).ReadBytes(data);  
var read = ProcessData(data);  
await Dispatcher.RunAsync(Windows.UI.Core.CoreDispatcherPriority.Normal, () =>  
{  
    txtHeartRate.Text = read.ToString();  
});
```

# demo

Heart Rate Monitor



# Background tasks

- 1. Creare un progetto Windows Phone Runtime component e implementare `IBackgroundTask`

```
public sealed class PolarH7Task : IBackgroundTask
{
    0 references
    public async void Run(Windows.ApplicationModel.Background.IBackgroundTaskInstance taskInstance)
    {
        BackgroundTaskDeferral deferral = taskInstance.GetDeferral();
        try
        {
            // ...
        }
    }
}
```

- 2. Aggiungere reference e modificare **Package.appxmanifest**

```
<Extensions>
  <Extension Category="windows.backgroundTasks" EntryPoint="PolarH7Background.PolarH7Task">
    <BackgroundTasks>
      <m3:Task Type="gattCharacteristicNotification" />
    </BackgroundTasks>
  </Extension>
</Extensions>
```

```
<!--Background tasks-->
<xs:simpleType name="ST_BackgroundTasks">
  <xs:restriction base="xs:string">
    <xs:enumeration value="chatMessageNotification"/>
    <xs:enumeration value="rfcommConnection"/>
    <xs:enumeration value="deviceConnectionChange"/>
    <xs:enumeration value="bluetoothSignalStrength"/>
    <xs:enumeration value="gattCharacteristicNotification"/>
  </xs:restriction>
</xs:simpleType>
```

# Background Tasks

- 3. Verificare se l'app ha il consenso di girare nel background

```
if (await BackgroundExecutionManager.RequestAccessAsync() == BackgroundAccessStatus.Denied)
{
    // TODO: What?
}
```

- 4. Registrare un nuovo task

```
GattCharacteristicNotificationTrigger trigger = new Windows.ApplicationModel.Background.GattCharacteristicNotificationTrigger(cHeart);
BackgroundTaskBuilder builder = new BackgroundTaskBuilder();
builder.Name = device.BluetoothAddress.ToString("x012");
builder.TaskEntryPoint = typeof(PolarH7Task).ToString();
builder.CancelOnConditionLoss = false;
```

```
builder.SetTrigger(trigger);
```

```
TaskRegistration = builder.Register();
```

- 4. Rimuovere/gestire i task registrati

```
// Unregister any remaining background tasks
foreach (BackgroundTaskRegistration reg in BackgroundTaskRegistration.AllTasks.Values)
    reg.Unregister(false);
```

# demo

Keep the keys + Heart Rate in Background



# Limitazioni API Windows Phone

- Non è possibile scollegarsi manualmente da un dispositivo
- Nessun indicazione per il campo RSSI
- Pre-pairing necessario per avere accesso dalle app = **NO BEACON**
- Le app hanno esclusività sui servizi – la prima app che si collega ad un servizio o registra un background task per un servizio ha Esclusività
  - Un'altra app che prova collegarsi riceverà access denied
  - L'altra app non sa chi ha accesso esclusivo quindi non può fornire indicazioni all'utente
- API solo Client Mode – Server Mode
  - Alcune API sono disponibile per OEM e dev Manufacturers



# Lumia SensorCore Beta



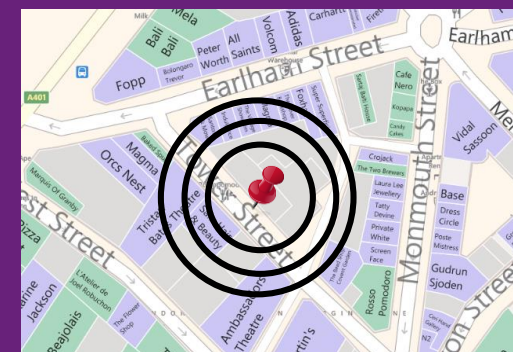
# Cos'è Lumia SensorCore?

- Collezione di API che usano i dati forniti da vari sensori: accelerometro, location
- Usato per tenere traccia delle attività fisica e movimento
- I sensori girano in background e raccolgono dati per un massimo di 10 giorni
- Sempre attivi ma in Low Power Mode
- Potrebbe fornire accesso a dati PRIVATI quindi può essere disabilitata e lo storico può essere cancellato



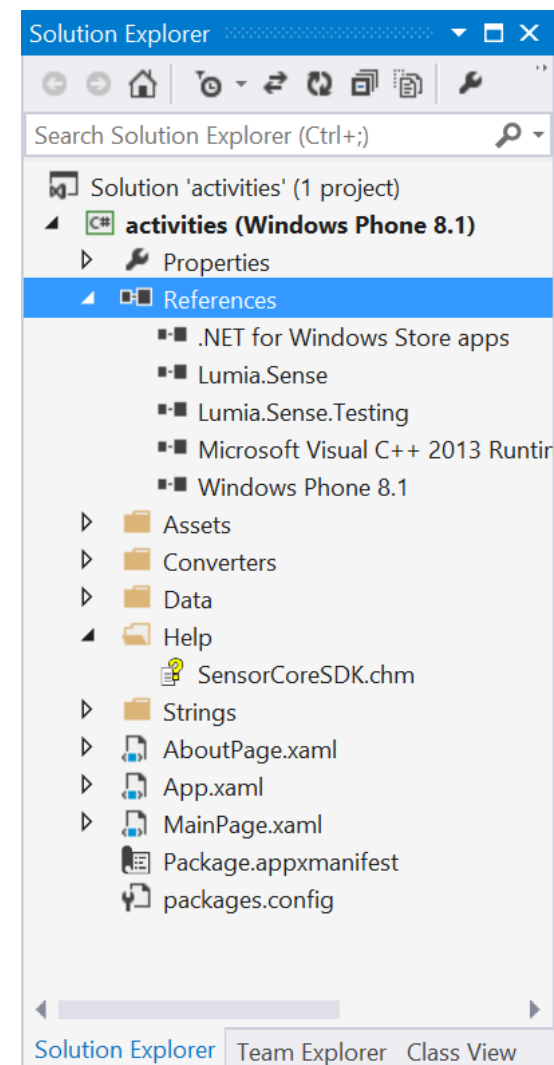
# API

- Step Counter
- Activity Monitor
- Place Monitor
- TrackPoint Monitor



# Visual Studio

- DLL disponibile da **NuGet**
- Architettura ARM o x86
- Microsoft Visual C++ 2013 Runtime Package for Windows Phone
- Testing Tools opzionale









# Capabilities

- Quando si installa l'SDK vengono aggiunte in automatico dentro **Package.appxmanifest**

```
<DeviceCapability Name="location" />
<m2:DeviceCapability Name="humaninterfacedevice">
  <m2:Device Id="vidpid:0421 0716">
    <m2:Function Type="usage:ffaa 0001" />
    <m2:Function Type="usage:ffee 0001" />
    <m2:Function Type="usage:ffee 0002" />
    <m2:Function Type="usage:ffee 0003" />
    <m2:Function Type="usage:ffee 0004" />
  </m2:Device>
</m2:DeviceCapability>
```

# API State








	Name	Description
	<a href="#">LocationEnabled</a>	<code>true</code> if the location setting of the phone is enabled, <code>false</code> otherwise.
	<a href="#">SenseEnabled</a>	<code>true</code> if the motion data setting of the phone is enabled, <code>false</code> otherwise.






	Name	Description
	<a href="#">GetApiStateAsync</a>	Returns Sense and Location API state
	<a href="#">GetSenseError</a>	Returns Sense error matching the given HRESULT code from an exception
	<a href="#">LaunchLocationSettingsAsync</a>	Launches Location settings
	<a href="#">LaunchSenseSettingsAsync</a>	Launches Sense settings

# Step Counter

- Quanti passi e per quanto tempo
- Caminata o corsa – usa anche l'intensità del movimento non solo la velocità
- Granularità 5 minuti – storico 10 giorni
- 5-6 secondi fino quando inizia vedere gli eventi
- Ci possono essere falsi positivi

# Step Counter API

	Name	Description
	<a href="#">ActivateAsync</a>	Reestablish the communication channel with underlying sensor, if not already exists
	<a href="#">DeactivateAsync</a>	Close the communication with underlying sensor, this explicitly closes the communication channel.
	<a href="#">GetCurrentReadingAsync</a>	Gets the current reading.
	<a href="#">GetDefaultAsync</a>	Gets the default implementation.
	<a href="#">GetStepCountAtAsync</a>	Gets the step count at given time.
	<a href="#">GetStepCountHistoryAsync</a>	Returns time ordered list of step counts during given time period. Data granularity is usually around five minutes.
	<a href="#">IsSupportedAsync</a>	Returns whether the sensor is supported by the device or not.








	Name	Description
	<a href="#">RunningStepCount</a>	Gets the number of running steps taken since the motion data was enabled.
	<a href="#">RunTime</a>	Gets the time spent running since the motion data was enabled.
	<a href="#">Timestamp</a>	Gets the creation time of the sensor reading.
	<a href="#">WalkingStepCount</a>	Gets the number of walking steps taken since the motion data was enabled.
	<a href="#">WalkTime</a>	Gets the time spent walking since the motion data was enabled.






# Activity Monitor

- Cambiamenti nell'attività del utente: fermo, cammina, corre...
- Delay tipico 5-10 sec per eliminare alcuni falsi positivi
- Risultati migliori dentro la borsa o in tasca
- Bisogna abilitare l'API
- API con risultati real-time e storico 10 giorni

# Activity Monitor API

	Name	Description
	<a href="#">ActivateAsync</a>	Reestablish the communication channel with underlying sensor, if not already exists
	<a href="#">DeactivateAsync</a>	Close the communication with underlying sensor, this explicitly closes the communication channel.
	<a href="#">GetActivityAtAsync</a>	Gets the device activity at given time.
	<a href="#">GetActivityHistoryAsync</a>	Returns time ordered list of activities occurred during given time period.
	<a href="#">GetCurrentReadingAsync</a>	Gets the current activity
	<a href="#">GetDefaultAsync</a>	Gets the default implementation.
	<a href="#">IsSupportedAsync</a>	Returns whether the sensor is supported by the device or not.









	Name	Description
	<a href="#">Enabled</a>	Enables or disables activity change event monitoring.
	<a href="#">Type</a>	The sensor type.





	Name	Description
	<a href="#">ReadingChanged</a>	Occurs each time activity changes.

# Place Monitor

- Lista di geo-coordinate
- Usa principalmente celle del operatore e/o hotspot WiFi. GPS solo se un'altra applicazione lo usa
- Non fornisce dati in tempo reale
- Prova indovinare l'indirizzi di casa e lavoro
- Classificazione può durare 2-3 giorni
- Area raggio almeno 200 metri, distanze minimo 500 m, tempo di permanenza almeno 10 minuti

# Place Monitor API







	Name	Description
	<a href="#">ActivateAsync</a>	Reestablish the communication channel with underlying sensor, if not already exists
	<a href="#">DeactivateAsync</a>	Close the communication with underlying sensor, this explicitly closes the communication channel.
	<a href="#">GetDefaultAsync</a>	Gets the default implementation.
	<a href="#">GetHomeAsync</a>	Gets the home location.
	<a href="#">GetKnownPlaceAsync</a>	Gets place by place id.
	<a href="#">GetKnownPlacesAsync</a>	Gets the set of currently known places.
	<a href="#">GetWorkAsync</a>	Gets the work location.
	<a href="#">IsSupportedAsync</a>	Returns whether the sensor is supported by the device or not.

	Name	Description
	<a href="#">Id</a>	Unique identifier of the place.
	<a href="#">Kind</a>	Type or kind of the place.
	<a href="#">Position</a>	Geographic position of the place.
	<a href="#">Radius</a>	The radius of the circular area of the place centered at <a href="#">Position</a> in meters.





# TrackPoint Monitor

- Simile a Place Monitor ma traccia le rute
- API offline come Place Monitor
- Track point ogni 5 minuti e un minimo di 500 metri
- Accuratezza dipende da numero di celle o hotspot. Una corsa in un parco senza una sessione GPS attiva potrebbe registrare un solo trackpoint

# TrackPoint Monitor API

	Name	Description
	<a href="#">ActivateAsync</a>	Reestablish the communication channel with underlying sensor, if not already exists
	<a href="#">DeactivateAsync</a>	Close the communication with underlying sensor, this explicitly closes the communication channel.
	<a href="#">GetDefaultAsync</a>	Gets the default implementation.
	<a href="#">GetPointAtAsync</a>	Returns the track point of the device at given time.
	<a href="#">GetTrackPointsAsync</a>	Returns the collected track points the device moved during the given time period.
	<a href="#">IsSupportedAsync</a>	Returns whether the sensor is supported by the device or not.

	Name	Description
	<a href="#">TrackPoint(TrackPoint)</a>	Constructor
	<a href="#">TrackPoint(BasicGeoposition, Double, TimeSpan, DateTime)</a>	Constructor

	Name	Description
	<a href="#">LengthOfStay</a>	Time how long the device stayed at this point.
	<a href="#">Position</a>	Geographic position of the track point.
	<a href="#">Radius</a>	The estimated radius of a circular area around the location which reflects the used positioning technology.
	<a href="#">Timestamp</a>	Time of entry to the location.

# **Ci saranno delle novità Lumia SensorCore Release e Lumia Denim**

# demo

<https://github.com/nokia-developer>





# Q&A

Tutto il materiale di questa sessione su

<http://www.communitydays.it/>

Lascia il feedback su questa sessione dal sito,  
potrai essere estratto per i nostri premi!

Seguici su

Twitter @CommunityDaysIT

Facebook <http://facebook.com/cdaysit>

#CDays14

