Orbiter4

1.1 Event Processors

A variety of lap starting, monitoring and laps counting schemes have been designed to allow you to run events. These fall into several general categories:

- Track layout: Loop, Start/Finish and Triathlon
  The positioning of your detectors on the course and the meaning of each detection
- Start: Chip Start, Gun Start, Wave Start, Multi-Start
  Start time determination: Chip Start events require a detection at the start line. Gun Start events give all participants the same start time. Wave Start events give participants the start time of each wave. Multi-Start events allow new event results to be created each time the participant starts.
- Validation: Waypoint
  Mid-course detections to ensure participant attained mid-course milestones

Track layouts are comprised of detectors located at positions on the track. Each position has a distance from the start of the track, a detector orientation and a minimum time which must be exceeded for the detection to be counted in the event.

- Multiple detectors may be located at each position to improve timing accuracy.
- Position 0 is the start position.
- Position *n* is the finish position on Start/Finish events.
- First tag detection seen:
  The detector antenna faces across the track. The first time the tag is detected in the detection field establishes the detection time.
- Last tag detection seen:
  The detector antenna faces up the track. The last time the tag is detected in the detection field establishes the detection time.
- Minimum Leg Time:
  To mitigate the possibility of impossible velocities being counted, a minimum amount of time since the previous detection can be set.
- Minimum Lap Time:
  To mitigate the possibility of impossible velocities being counted, a minimum amount of time for a lap can be set. This time is also used for Gun Start Loop events so that detections at the start/finish line are ignored at the start of the event.
- Maximum Lap Time:
  Multi-Start events use this time to determine when a participant has stopped running on the track for the current session.

When running a wave event, participants can either have assigned waves or can register for a wave on the course.

- Assigned Waves:
  The operator assigns the wave for each participant and his start time is set when the

assigned wave starts.

- Course Registered Waves:
  The participant presents his tag to the registration detector. The participant is then assigned to the next wave and will have that wave's start time.

1.1.1 Loop Chip Start

Simple loop track with detectors located at the loop start/finish line. Additional detectors may be placed along track for event monitoring.

- Position 0, start and finish position, required
- Positions 1..*n*, optional, midpoint positions, only used to update event viewers

Once the event starts, each participant must be detected at position 0 to begin his race, which sets his start time. Once he has started his race, laps will be counted each time he crosses position 0.

1.1.2 Start/Finish Chip Start

Simple start line and finish line track with detectors located at the start line and detectors located at the finish line.

- Position 0, start position, required
- Positions 1..*n-1*, optional, midpoint positions, only used to update event viewers
- Position *n*, finish position, required

Once the event starts, each participant must be detected at position 0 to begin his race, which sets his start time.

Once he has started his race, laps will be counted each time he crosses position *n*.

1.1.3 Triathlon Chip Start with 3 detectors

Triathlon with detectors located at the start line and later moved to the finish line, and detectors located at the transition zone entrance and the transition zone exit. The entrance is used for the bike and run transitions.

- Position 0, start and finish position, required
- Position 1, bike transition, required
- Position 2, run transition, required

Once the event starts, each participant must be detected at position 0 to begin his race, which sets his start time.

The participant must be detected at position 1, the first time he is detected at position 1, his swim time is established. The participant is expected to be detected a second time at position 1 to establish the swim to bike transition time. If no second detection is made at position 1, the transition time becomes part of the bike time.

The participant must be detected at position 2, the first time he is detected at position 2, his bike time is established. The participant is expected to be detected a second time at position 2 to establish the bike to run transition time. If no second detection is made at position 2, the transition time becomes part of the run time.

The participant must be detected at position 0 to complete the race and to establish his run time.

1.1.4 Triathlon Chip Start with 3 detectors, ABBA

Triathlon with detectors located at the start line and later moved to the finish line, and detectors located at the transition zone entrance and the transition zone exit. The transition zone is entered in one direction for the bike transition and in the opposite direction for the run transition.

- Position 0, start and finish position, required
- Position 1, bike transition in, run transition out, required
- Position 2, run transition in, bike transition out, required

Once the event starts, each participant must be detected at position 0 to begin his race, which sets his start time.

The participant must be detected at position 1, the first time he is detected at position 1, his swim time is established. The participant is then expected to be detected at position 2 within a certain number of minutes to establish the swim to bike transition time. If no detection is made at position 2, the transition time becomes part of the bike time.

The participant must be detected at position 2, the first time he is detected at position 2, his bike time is established. The participant is then expected to be detected at position 1 within a certain number of minutes to establish the bike to run transition time. If no second detection is made at position 1, the transition time becomes part of the run time.

The participant must be detected at position 0 to complete the race and to establish his run time.

1.1.5 Loop Gun Start

- Position 0, start and finish position, required
- Positions 1..*n*, optional, midpoint positions, only used to update event viewers

Once the event starts, each participant will be given the start time of the event as his start time. Any detection at position 0 prior to ½ of the Maximum Lap Time is ignored as they are likely from the participant starting the race. Laps will be counted each time he crosses position 0.

1.1.6 Start/Finish Gun Start

- Position 0, start position, ignored
- Positions 1..*n-1*, optional, midpoint positions, only used to update event viewers
- Position *n*, finish position, required

Once the event starts, each participant will be given the start time of the event as his start time. Any detection at position 0 is ignored as they are from the participant starting the race. Laps will be counted each time he crosses position *n*.

1.1.7 Triathlon Gun Start with 3 detectors

Triathlon with detectors located at the start line and later moved to the finish line, and detectors located at the transition zone entrance and the transition zone exit. The entrance is used for the bike and run transitions.

- Position 0, finish position, required
- Position 1, bike transition, required
- Position 2, run transition, required

Once the event starts, each participant has his start time set to the event start time.

The participant <u>must</u> be detected at position 1, the first time he is detected at position 1, his swim time is established. The participant is expected to be detected a second time at position 1 to establish the swim to bike transition time. If no second detection is made at position 1, the transition time becomes part of the bike time.

The participant <u>must</u> be detected at position 2, the first time he is detected at position 2, his bike time is established. The participant is expected to be detected a second time at position 2 to establish the bike to run transition time. If no second detection is made at position 2, the transition time becomes part of the run time.

The participant must be detected at position 0 to complete the race and to establish his run time.

1.1.8 Triathlon Gun Start with 3 detectors, ABBA

Triathlon with detectors located at the start line and later moved to the finish line, and detectors located at the transition zone entrance and the transition zone exit. The transition zone is entered in one direction for the bike transition and in the opposite direction for the run transition.

- Position 0, finish position, required
- Position 1, bike transition in, run transition out, required
- Position 2, run transition in, bike transition out, required

Once the event starts, each participant has his start time set to the event start time.

The participant <u>must</u> be detected at position 1, the first time he is detected at position 1, his swim time is established. The participant is then expected to be detected at position 2 within a certain number of minutes to establish the swim to bike transition time. If no detection is made at position 2, the transition time becomes part of the bike time.

The participant <u>must</u> be detected at position 2, the first time he is detected at position 2, his bike time is established. The participant is then expected to be detected at position 1 within a certain number of minutes to establish the bike to run transition time. If no second detection is made at position 1, the transition time becomes part of the run time.

The participant must be detected at position 0 to complete the race and to establish his run time.

1.1.9 Loop Wave Start

- Position 0, finish position, required; wave registration if using course registered waves

- Positions 1..*n*, optional, midpoint positions, only used to update event viewers

As each wave starts, each participant assigned to the wave or course registered for the wave will be given the start time of the wave as his start time. Any detection at position 0 for participants that have not already started will be course registered to the next wave start. Laps will be counted each time he crosses position 0.

1.1.10 Start/Finish Wave Start

- Position 0, registration position
- Positions 1..*n-1*, optional, midpoint positions, only used to update event viewers
- Position *n*, finish position, required

As each wave starts, each participant assigned to the wave or course registered for the wave will be given the start time of the wave as his start time. Any detection at position 0 for participants that have not already started will be course registered to the next wave start. Laps will be counted each time he crosses position *n*.

1.1.11 Triathlon Wave Start with 3 detectors

Triathlon with detectors located at the start line and later moved to the finish line, and detectors located at the transition zone entrance and the transition zone exit. The entrance is used for the bike and run transitions.

- Position 0, finish position, required; wave registration if using course registered waves
- Position 1, bike transition, required
- Position 2, run transition, required

As each wave starts, each participant assigned to the wave or course registered for the wave will be given the start time of the wave as his start time. Any detection at position 0 for participants that have not already started will be course registered to the next wave start.

The participant must be detected at position 1, the first time he is detected at position 1, his swim time is established. The participant is expected to be detected a second time at position 1 to establish the swim to bike transition time. If no second detection is made at position 1, the transition time becomes part of the bike time.

The participant must be detected at position 2, the first time he is detected at position 2, his bike time is established. The participant is expected to be detected a second time at position 2 to establish the bike to run transition time. If no second detection is made at position 2, the transition time becomes part of the run time.

The participant must be detected at position 0 to complete the race and to establish his run time.

1.1.12 Triathlon Wave Start with 3 detectors, ABBA

Triathlon with detectors located at the start line and later moved to the finish line, and detectors located at the transition zone entrance and the transition zone exit. The transition zone is entered in one direction for the bike transition and in the opposite direction for the run transition.

- Position 0, finish position, required; wave registration if using course registered waves
- Position 1, bike transition in, run transition out, required
- Position 2, run transition in, bike transition out, required

As each wave starts, each participant assigned to the wave or course registered for the wave will be given the start time of the wave as his start time. Any detection at position 0 for participants that have not already started will be course registered to the next wave start.

The participant <u>must</u> be detected at position 1, the first time he is detected at position 1, his swim time is established. The participant is then expected to be detected at position 2 within a certain number of minutes to establish the swim to bike transition time. If no detection is made at position 2, the transition time becomes part of the bike time.

The participant <u>must</u> be detected at position 2, the first time he is detected at position 2, his bike time is established. The participant is then expected to be detected at position 1 within a certain number of minutes to establish the bike to run transition time. If no second detection is made at position 1, the transition time becomes part of the run time.

The participant must be detected at position 0 to complete the race and to establish his run time.

1.1.13 Loop Multi-Start

Static track set up to allow participants to come to the course at any time and run some timed laps.

- Position 0, start and finish position, required
- Positions 1..*n*, optional, midpoint positions, used to verify participant is running course and to update event monitors

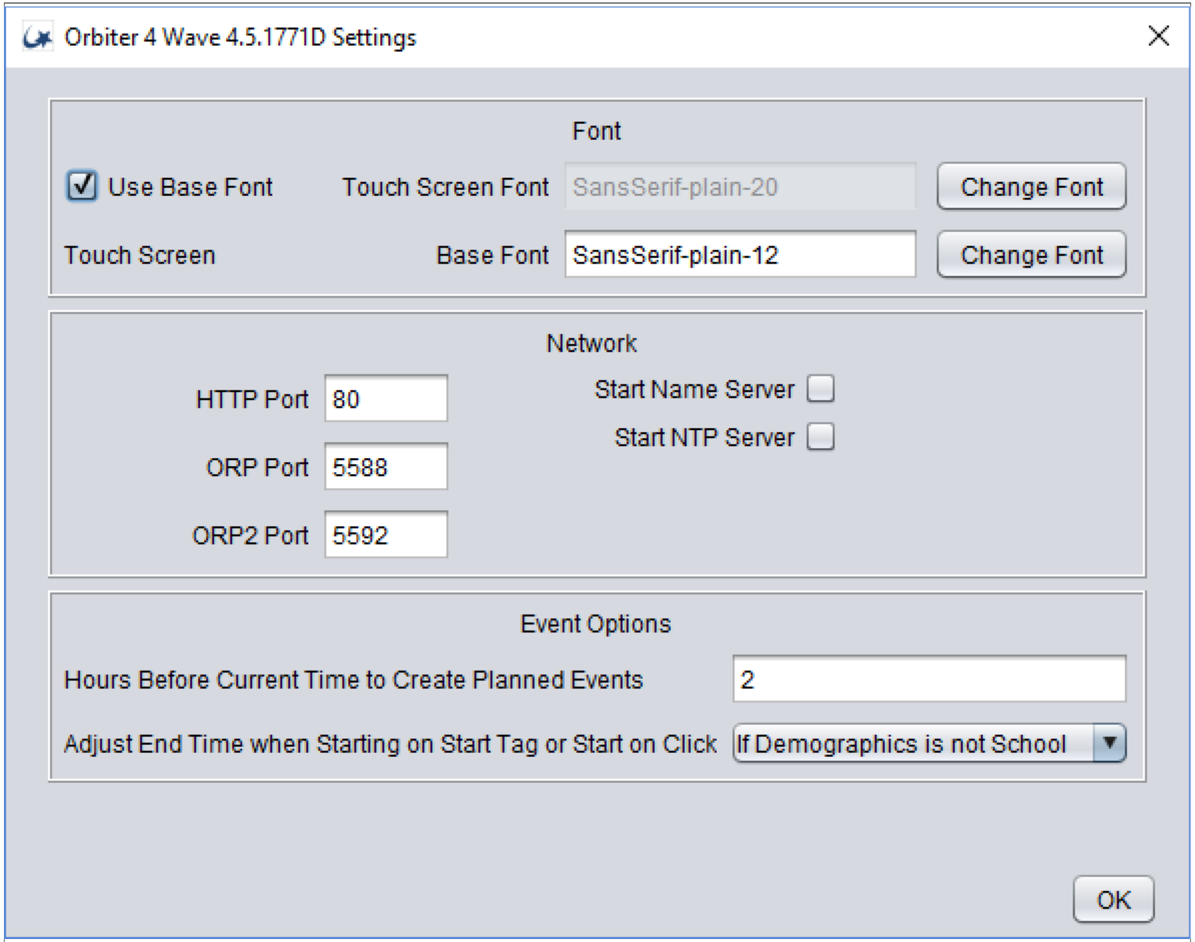
To start a timed lap session, each participant <u>must</u> be detected at position 0 to begin the session, which sets his session start time. Once he has started his session, laps will be counted each time he crosses position 0. When a participant has not been detected at position 0 for Maximum Lap Time, the timed session is ended and a new session may be started on his next visit.

1.2 Web Interface

On startup, Orbiter4 open an HTTP port to accept HTTP requests. The web processor utilizes *templar* to allow the page designer to access data and present data.

1.2.1 HTTP Port

The Web Interface opens the port 80 socket to accept HTTP requests.


Orbiter 4 Wave 4.5.1771D Settings

Font

☑ Use Base Font — Touch Screen Font: SansSerif-plain-20 — Change Font

Touch Screen — Base Font: SansSerif-plain-12 — Change Font

Network

HTTP Port: 80
ORP Port: 5588
ORP2 Port: 5592
Start Name Server ☐
Start NTP Server ☐

Event Options

Hours Before Current Time to Create Planned Events: 2
Adjust End Time when Starting on Start Tag or Start on Click: If Demographics is not School

OK

Illustration 1: HTTP Web Port Selection

1.2.2 Root

HTML files are rooted at *orbiter-home*\var\http\orbiter

1.2.3 Data Access Through RestFUL

Access to data is provided using a RestFUL interface. Refer to Section A for syntax.

1.2.4 Data Access Through Templar

Access to data in Orbiter4 is provided by a tool called Templar. Refer to Section B for syntax and usage of Templar.

Type	Properties	Type	Notes
global	orbiter	Orbiter	
	session	Session	
Orbiter	activeEvents	ActiveEvent[]	Array of events actively running
	resultsByBibNumber(event-id)	Event	Sorted by bib number
	resultsByParticipant(event-id)	Event	Sorted by participant last, first

Type	Properties	Type	Notes
ActiveEvent	currentResults	EventResult[]	Sorted by place
	currentWave	Integer	
	detectionsCollection	Detection[]	
	event	Event	
Event	active	Boolean	
	description	String	
	endTimeAsLong	Long	
	eventId	Integer	
	eventStatus	EventStatus	
	eventType	EventType	
	fullname	String	
	laps	Integer	
	masterEvent	Event	
	name	String	
	startTimeAsLong	Long	
	teamResultsCollection	TeamResult[]	
	track	Track	
EventResult	distance	String	Participant last name
	elapsedTimeAsLong	Long	
	endTimeAsLong	Long	
	lapCount	Integer	
	participant	Participant	
	place	Integer	
	startTimeAsLong	Long	
Participant	active	Boolean	
	age	Integer	years
	city	String	
	dob	Date	
	firstname	String	
	fullname	String	
	gender	String	
	height	String	
	lastname	String	
	middleInitiall	String	

Type	Properties	Type	Notes
	state	String	
Track	active	Boolean	
	description	String	
	lastPosition	Integer	
	maxLapTimeAsLong	Long	Milliseconds
	minLapTimeAsLong	Long	Milliseconds
	name	String	
	trackName	String	
	venueName	String	
	trackPositionsList	TrackPosition[]	
Session	parms[key]	Value	Parameter from URL
TrackPosition	distanceFromStart	Double	
	minLegTimeAsLong	Long	
	name	String	
	position	Integer	
	track	Track	

```
Orbiter
  activeEvents[]
    event
      name

session
  parms[]
```

Illustration 2: Templar variables

```
<!doctype html>
<html lang="en">
  <head>
    <title>Orbiter 4 Wave</title>
    <link rel="stylesheet" href="/static/css/reset.css" />
    <link rel="stylesheet" href="/static/css/style.css" />
    <!-- <script src="//ajax.googleapis.com/ajax/libs/jquery/2.0.3/jquery.min.js"></script> -->
    <link rel="icon" href="/static/images/Orbiter.png" type="image/png" />
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
  </head>
  <body>
    <img src="static/images/Orbiter.gif"/><br>

    <div id="main">
      <h1>Active Events</h1>
      <table>
      {loop orbiter.activeEvents as activeEvent}
      <tr>
        <td>Event {activeEvent.event.name}</td>
        <td><a href="orbiter/eventParticipant.html.templar?eventId={activeEvent.eventId}">
            By Participant</a>
        </td>
        <td><a href="orbiter/eventBibNumber.html.templar?eventId={activeEvent.eventId}">
            By BibNumber</a>
        </td>
      </tr>
      {endloop}
      </table>
    </div>
  </body>
</html>
```

Illustration 3: index.html.templar

A RestFUL

A.1 Queries

A.1.1 /events/

```
/events
/events/{event-id|active}
/events/{event-id|active}/results
/events/{event-id|active}/results/result-id
/events/{event-id|active}/teamResults
/events/{event-id|active}/teamResults/result-id
/events/{event-id|active}/participants
/events/{event-id|active}/participants/participant-id
/events/{event-id|active}/detections
/events/{event-id|active}/detections/detection-id
```

- /events/
  Return JSON list of events
- /events/*event-id*
  /events/active
  Return JSON object of specified event.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
- /events/*event-id*/results
  /events/active/results
  Return JSON array of event results.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
- /events/*event-id*/results/*result-id*
  /events/active/results/*result-id*
  Return JSON object of specified event results.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
  If *result-id* does not exist, 404 (not found) is returned.
- /events/*event-id*/teamResults
  /events/active/teamResults
  Return JSON array of team event results.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
- /events/*event-id*/teamResults/*team-result-id*
  /events/active/results/*team-result-id*
  Return JSON object of specified team event result.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
  If *team-result-id* does not exist, 404 (not found) is returned.

- /events/*event-id*/detections
  /events/active/detections
  Return JSON array of event detections.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.

- /events/*event-id*/detections/*detection-id*
  /events/active/detections/*detection-id*
  Return JSON object of specified detection.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
  If *detection-id* does not exist, 404 (not found) is returned.

- /events/*event-id*/participants
  /events/active/participants
  Return JSON array of event participants.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.

- /events/*event-id*/participants/*participant-id*
  /events/active/participants/*participant-id*
  Return JSON object of specified participant.
  If *event-id* does not exist or there is no active event, 404 (not found) is returned.
  If *participant-id* does not exist, 404 (not found) is returned.

A.1.2 /results/*result-id*/detections

```
/results/result-id/detections
/results/result-id/detections/detection-id
```

- /results/*result-id*/detections
  Return JSON array of specified detections.
  If *result-id* does not exist, 404 (not found) is returned.

- /results/*result-id*/detections/*detection-id*
  Return JSON object of specified detections.
  If *result-id* does not exist, 404 (not found) is returned.
  If *detection-id* does not exist, 404 (not found) is returned.

A.1.3 /participants

```
/participants
/participants/participant-id
```

- /participants
  Return JSON array of participants.

- /participants/*participant-id*
  Return JSON object of specified participant.
  If *participant-id* does not exist, 404 (not found) is returned.

A.1.4 /tracks

```
/tracks
/tracks/track-id
```

- /tracks

Return JSON array of tracks.

- /tracks/*track-id*
  Return JSON object of specified track.
  If *track-id* does not exist, 404 (not found) is returned.

A.2 Objects

A.2.1 Event

```
{
    "eventId": 1000,
    "venue": "Test Venue",
    "distance": 100,
    "trackType": "Loop",
    "_links": {
        "teamResults": {
            "href": "/events/1000/teamResults/"
        },
        "childEvents": [],
        "results": {
            "href": "/events/1000/results/"
        },
        "tracks": {
            "href": "/tracks/1000"
        },
        "detections": {
            "href": "/events/1000/detections/"
        },
        "participants": {
            "href": "/events/1000/participants/"
        }
    },
    "description": "This is a test event.",
    "laps": 5,
    "startTimeAsLong": 1567356578720,
    "eventType": "Chip Start",
    "oganization": "Local Organization",
    "startInterval": 0,
    "endTimeAsLong": 1567357508720,
    "eventStatus": "Closed",
    "name": "Chip Start test 771",
    "startTime": "Sun, 01 Sep 2019 10:49:38 GMT",
    "endTime": "Sun, 01 Sep 2019 11:05:08 GMT",
    "track": "Test 771"
}
```

A.2.2 Result

```
{
    "endSplitTimeAsLong": 9130,
    "distance": 500,
    "_links": {
        "detection": {
            "href": "/results/1000/detections/"
        },
        "participant": {
            "href": "/events/1000/participants/1026"
        }
    },
    "eventResultId": 1000,
    "bibNumber": "771-1",
    "lapCount": 5,
    "startTimeAsLong": 1567356585080,
    "startPosition": "Position 0",
    "participant": "Katheleen Tesoriero",
    "elapsedTimeAsLong": 46210,
    "endPosition": "Position 0",
    "endTimeAsLong": 1567356631290,
    "startTime": "Sun, 01 Sep 2019 10:49:45 GMT",
    "endTime": "Sun, 01 Sep 2019 10:50:31 GMT",
    "place": 1,
    "endSplitTime": "00:00:09.13",
    "elapsedTime": "00:00:46.21"
}
```

A.2.3 TeamResult

```
    {
        "top8ElapsedTime": "No time",
        "distance": 2500,
        "top3ElapsedTime": "00:02:18.63",
        "top1ElapsedTime": "00:00:46.21",
        "lapCount": 25,
        "top6ElapsedTime": "No time",
        "N": 3,
        "top2ElapsedTime": "00:01:32.42",
        "teamResultId": 1000,
        "top7ElapsedTime": "No time",
        "top4ElapsedTime": "00:03:04.84",
        "top5ElapsedTime": "00:03:51.06",
        "topNElapsedTime": "00:02:18.63",
        "place": 2,
        "elapsedTime": "00:03:51.06"
    }
```

A.2.4 Detection

```
    {
        "manualEntry": false,
        "_links": {
            "participant": {
                "href": "/events/1000/participants/1026"
            }
        },
        "detectionTimeAsLong": 1567356585080,
        "position": "Position 0",
        "detectionTime": "Sun, 01 Sep 2019 10:49:45 GMT",
        "participant": "Katheleen Tesoriero",
        "detectionId": 1000
    }
```

A.2.5 Participant

```
    {
        "participantId": 1026,
        "firstName": "Katheleen",
        "lastName": "Tesoriero",
        "teams": [
            "Red Star Belgrade 0 771"
        ],
        "city": "Anywhere, USA",
        "fullName": "Katheleen Tesoriero",
        "age": 21,
        "tags": [
            {
                "userLabel": "L771-1",
                "tagId": 1021,
                "bibNumber": "771-1",
                "rfid": "771f00001"
            }
        ]
    }
```

A.2.6 Track

```
    {
        "venueName": "Test Venue",
        "distance": 100,
        "maxLapTime": "00:00:00.00",
        "trackId": 1000,
        "description": "This is a test track",
        "minLapTimeAsLong": 4000,
        "trackName": "Test 771",
        "maxLapTimeAsLong": 0,
        "minLapTime": "00:00:04.00",
        "tags": [
            {
                "orientation": "First tag detection seen",
                "positionId": 1000,
                "minLegTime": "00:00:04.00",
                "name": "Position 0",
                "detectors": "Loop 771",
                "minLegTimeAsLong": 4000,
                "distanceFromStart": 0
            }
        ]
    }
```

B Templar

Template provides an alternate means of returning event results.

B.1 Syntax Reference

In general all syntax starts with a {character and ends with the next available} character. This will either be an evaluation or a command, if it is not a command, templar will attempt to evaluate the expression.

Command Quick links

- `{-- ...`
- `{if ...`
- `{set ...`
- `{loop ...`
- `{import ...`
- `{static ...`
- `{statichtml ...`
- `{pre ...`
- `{tab}` or `{\t}`
- `{nl}` or `{\n}`
- `{dumpcontext}`
- `{dumpfunctions}`
- `{requires}`

B.2 EVALUATIONS

B.2.1 Method evaluations

```
{<evaluate>.<method>.<anotherMethod>}
```

Evaluations are the simplest of the templar syntax, it will do the following:

1. Lookup the <evaluate> in the templarContext
2. Call the <method> method on the <evaluate> object
3. Call the <anotherMethod> method on the object returned from the <evaluate>.<method> call
4. Render the above output

   Note that for each of the methods that is trying to be evaluated, the following prefixes will be looked up: get, is, has and finally just the method.

   For example, if the method to be evaluated is {person.name}, the person object will be looked up in the context, retrieved (if it exists), and then the following method lookups will be attempted (in order):

1. getName,
2. isName,
3. hasName, and finally
4. name

B.2.2 Function evaluations

You may also evaluate functions (see the function reference for a list of functions available):

```
{fn:size[<array>]}
```

1. Lookup the <array> in the templarContext
2. Render the size or length of the object

   You may also evaluate objects where the actual name of the object is not known until runtime (i.e. render time). In a rather contrived example, let's say that you wanted to loop through all of the objects in a session and print out the key:value pairs. In this case we use the dollar ($) notation:

```
{loop session.sessionObjects.keySet as key}
        {\t}KEY: {key}, VALUE: {session.sessionObjects.$key}{\n}
{endloop}
```

In effect this looks for a $key object in the templarContext with the name of key which is then called on the previous object.

B.3 COMMANDS

B.3.1 {-- ... - comments

Comments start with {-- and continue up to the first instance found of }.This means that you CANNOT have a } character in the comment section - sorry.

```
{--
```

We all know that comments in code are a good idea.

```
Comments can be multi-lined or they could be all on one line.  The only
character that is not allowed is the '}' character - as this ends the comment.

Apologies
---------

We understand that not allowing the end brace character in a comment
means that you cannot comment out large block of templar templates,
hence the apology.
```

```
}
```

B.3.2 {if ... - condition

```
{if <something>}
        <something> is true and this will be output
{else}
        <something> is false and this will be output
{endif}
```

<something> can either be a function, or an evaluation, e.g.:

```
{if fn:true[object.method]}
        fn:true[object.method] is true and this will be output
{else}
        fn:true[object.method] is false and this will be output
{endif}

{if some.object.methodThatReturnsABoolean}
        some.object.methodThatReturnsABoolean is true and this will be output
{else}
        some.object.methodThatReturnsABoolean is false and this will be output
{endif}
```

You may of course chain functions, so long as the result is coercible to a Boolean

```
{if fn:equals[fn:size[array], 10]}
        this will be output if the size of the array is exactly 10
{else}
        this will be output if the size of the array is not exactly 10
{endif}
```

B.3.3 {set ...} - set a variable into the context

The format of this command is as follows: {set___something___as somethingElse_}, where __something__is what you want to place in the context, and___somethingElse___is the key which it will be bound to.

```
{set "world" as hello}
```

We will be using this example later on - you can see that it has been placed into the context by using the {dumpcontext} command - which renders the following (at runtime):

```
TemplarContext[{header-h1:templar}, {header-tagline:an easy to use, ultra light-
weight, templating engine}, {header-h2:Welcome to templar}, {github-
org:synapticloop}, {hello:world}, {sidebar-h3:templar}, {head-title:templar by
synapticloop}, {github-repo:templar}, {sidebar-tagline:an easy to use, ultra light-
weight, templating engine}]
```

You can also set the evaluation of an object or function into the context.

```
{set fn:length["some-string"] as someStringLength}
```

Once again dumping the context

```
TemplarContext[{header-h1:templar}, {header-tagline:an easy to use, ultra light-
weight, templating engine}, {header-h2:Welcome to templar},{someStringLength:11},
{github-org:synapticloop}, {hello:world}, {sidebar-h3:templar},{head-title:templar
by synapticloop}, {github-repo:templar}, {sidebar-tagline:an easy to use, ultra
light-weight, templating engine}]
```

B.3.4 {loop ...} - loop over an iterable

The format of this command is as follows: {loop___something___as_somethingElse_}, where __something__is what you want to loop over, placing___somethingElse___into the context for each iteration. This will iterate over it rendering the inner content between start and {endloop} token.

```
{loop object.array as something}
  // render something here
{endloop}
```

An example taken from h2zero:

```
{loop table.finders as finder}
        {if fn:null[finder.selectClause]}
                {\t}private static final String SQL_{finder.staticName} =
SQL_SELECT_START
        {else}
                {\t}private static final String SQL_{finder.staticName} =
"{finder.selectClause}"
        {endif}

        {if fn:notNull[finder.whereClause]} + " {finder.whereClause}"{endif}
        {if fn:notNull[finder.orderBy]} + " order by {finder.orderBy}"{endif};{\n}
{endloop}
{\n}
```

B.3.5 {import ...} - import a file into the current one

Import a file into the current file and parse the file for any templar declarations. The file to import can also be from the classpath (if you are deploying this as a complete jar - {import classpath:/some/classpath/file/here.templar}

```
{import src/imports/commands/import.templar}
```

This file was imported from syntax-reference.html with the above templar declaration.

B.3.6 {static ...} - import a file into the current one

Import a file into the current file and DO NOT parse the file for any templar declarations. The file to import can also be from the classpath (if you are deploying this as a complete jar - {static classpath:/some/classpath/file/here.templar}

```
{import src/imports/commands/static.templar}
```

This file was imported from syntax-reference.html with the above line

B.3.7 {statichtml ...} - import a file into the current one and escape HTML characters

Import a file into the current file and DO NOT parse the file for any templar declarations. The file to import can also be from the classpath (if you are deploying this as a complete jar -

{statichtml classpath:/some/classpath/file/here.templar} Escape any characters for HTML, i.e.:

- & is escaped to &amp;
- < is escaped to &lt;
- > is escaped to &gt;

```
{import src/imports/commands/static.templar}
```

This file was imported from syntax-reference.html with the above line

B.3.8 {pre ... - pre formatted text

As the name would suggest pre-formatted text - this is the only command that does not end with a }character, instead must end with pre}.

```
{pre
```

Anything that goes in here, including any templar syntax is output exactly asshown without any templar parsing. This is especially useful for CSS and JavaScript which has a lot of ' and ' characters.

```
The content will be output until an end token of 'pre' immediately followed be a
'}' character

pre}
```

The above would render as follows (note the use of the {\n}):

```
{\n}
{\n}
{\n}

anything that goes in here, including any templar syntax is output exactly as shown
without any templar parsing.  This{\n}
is especially useful for CSS and JavaScript which has a lot of '{' and '}'
characters.{\n}
{\n}
The content will be output until an end token of 'pre' immediately followed be a
'}' character{\n}
{\n}
```

B.3.9 {tab} or {\t} - output a tab

This will output a tab whitespace character

B.3.10 {nl} or {\n} - output a new line character

This will output a newline whitespace character

B.3.11 {dumpcontext} - dumping the context

This is useful for debugging as this will render all of the objects in the context, and will look like this:

```
TemplarContext[{header-h1:templar}, {header-tagline:an easy to use, ultra light-
weight, templating engine}, {header-h2:Welcome to templar},{someStringLength:11},
{github-org:synapticloop}, {hello:world}, {sidebar-h3:templar},{head-title:templar
by synapticloop}, {github-repo:templar}, {sidebar-tagline:an easy to use, ultra
light-weight, templating engine}]
```

We cheated a little bit and added something to the context with the following:

```
{set "world" as hello}
```

B.3.12 {dumpfunctions} - dumping the functions

This is useful for debugging as this will render all of the registered functions and it will look like this:

```
FunctionLessThanEqual fn:<=[ <#numArgs: 2> ] aliased as (lte)
```

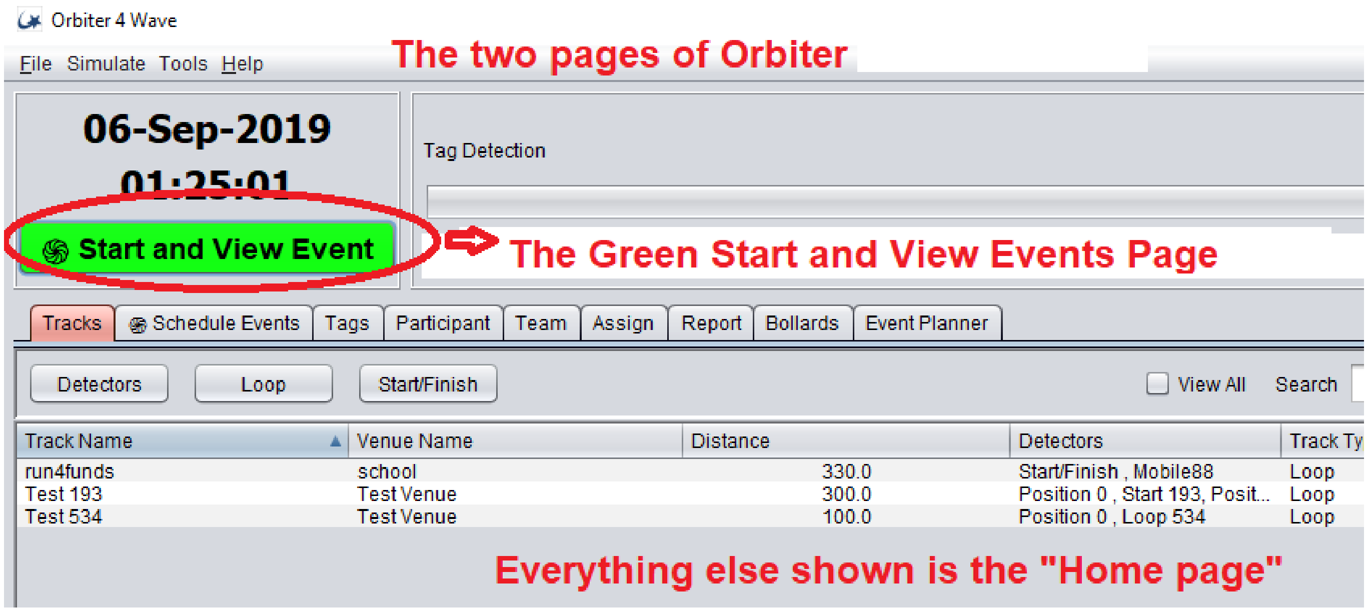
```
FunctionNotEqual fn:<>[ <#numArgs: 2> ] aliased as (notEqual, ne, not=)
FunctionInstanceOf fn:instanceOf[ <#numArgs: 2> ]
FunctionAnd fn:and[ <#numArgs: 2> ]
FunctionIndexOf fn:indexOf[ <#numArgs: 2> ]
FunctionPower fn:^[ <#numArgs: 2> ]
FunctionOr fn:or[ <#numArgs: 2> ]
FunctionIsNotNull fn:notNull[ <#numArgs: 1> ] aliased as (isNotNull, !null,
isnotnull, notnull, !Null)
FunctionFormatDate fn:fmtDate[ <#numArgs: 1 or 2> ]
FunctionModulus fn:%[ <#numArgs: 2> ]
FunctionAnd fn:&[ <#numArgs: 2> ]
FunctionLength fn:length[ <#numArgs: 1> ] aliased as (size)
FunctionFalse fn:false[ <#numArgs: 1> ]
FunctionMultiply fn:*[ <#numArgs: 2> ]
FunctionAdd fn:+[ <#numArgs: 2> ]
FunctionSubtract fn:-[ <#numArgs: 2> ]
FunctionOdd fn:odd[ <#numArgs: 1> ]
FunctionDivide fn:/[ <#numArgs: 2> ]
FunctionToJson fn:toJson[ <#numArgs: 1> ]
FunctionIsNull fn:null[ <#numArgs: 1> ] aliased as (isnull, isNull)
FunctionEven fn:even[ <#numArgs: 1> ]
FunctionTrue fn:true[ <#numArgs: 1> ]
FunctionLessThan fn:<[ <#numArgs: 2> ] aliased as (lt)
FunctionOr fn:|[ <#numArgs: 2> ]
FunctionEqual fn:=[ <#numArgs: 2> ] aliased as (eq, equal)
FunctionGreaterThan fn:>[ <#numArgs: 2> ] aliased as (gt)
FunctionGreaterThanEqual fn:>=[ <#numArgs: 2> ] aliased as (gt=)
```

B.3.13 {requires} - require a variable to be available in the context

This ensures that a variable is available in the context. If the variable is not available, then a RenderException will be thrown

C.1 Over View: Explanation of Orbiter4 from a visual GUI perspective.

This section explains the program from a Graphical Interface view. In a grand view of the software there are only two pages of importance. From these all Orbiter controls are gained. They are the "Home Page" and the Green "Start and View Events" page.



Track Name	Venue Name	Distance	Detectors	Track Ty
run4funds	school	330.0	Start/Finish , Mobile88	Loop
Test 193	Test Venue	300.0	Position 0 , Start 193, Posit...	Loop
Test 534	Test Venue	100.0	Position 0 , Loop 534	Loop

Our explanation begins on the Home Page tabs. They are located ¼ page down, and beigin on the left of the screen.   The user dills out the tabs from left to right to set up an event. Once the event is set up, then the event may be started from the Green “Start and View” events area.

An upper level explanation of the logic of the program in lay terms is as follows: the program is laid out as if a timing event is an invitation to a party. Like an invitation to a party, certain minimum information is needed to inform guests about the party. Such as location, time of the party, type of the party, invitations to the party, participants of the party, assigning and sending invitations to the party, and then once the party has started monitoring of the party, and finally results of the party. Consider the word party the same as Events. Each of the tabs moving from left to right is representative of the above. Tracks tab is filled out and identifies where the event is held. Schedule Events is when the event is held. Tags are like the RFID invitations to the event. Participants are those going to the event. Assign is how the tags are assigned to the participants. Bollards are where condition monitoring in real time with detectors to ensure the event is running properly. Event Scheduler is for operating many events at the same time.

The software allows for short cuts including importing csv files of mass numbers of participants quickly.   These are found in the far upper left of the screen under “Files”. Sample events are found under “Simulator” that shows the configuration of the software for basic events.   If there is a question on how to set up an event, it is best to run a simulation and look at how say a 5K event is set up. There are also tools, connectivity options, and the ability to back up the database. It’s always wise to back up your work so you can recover a set up.

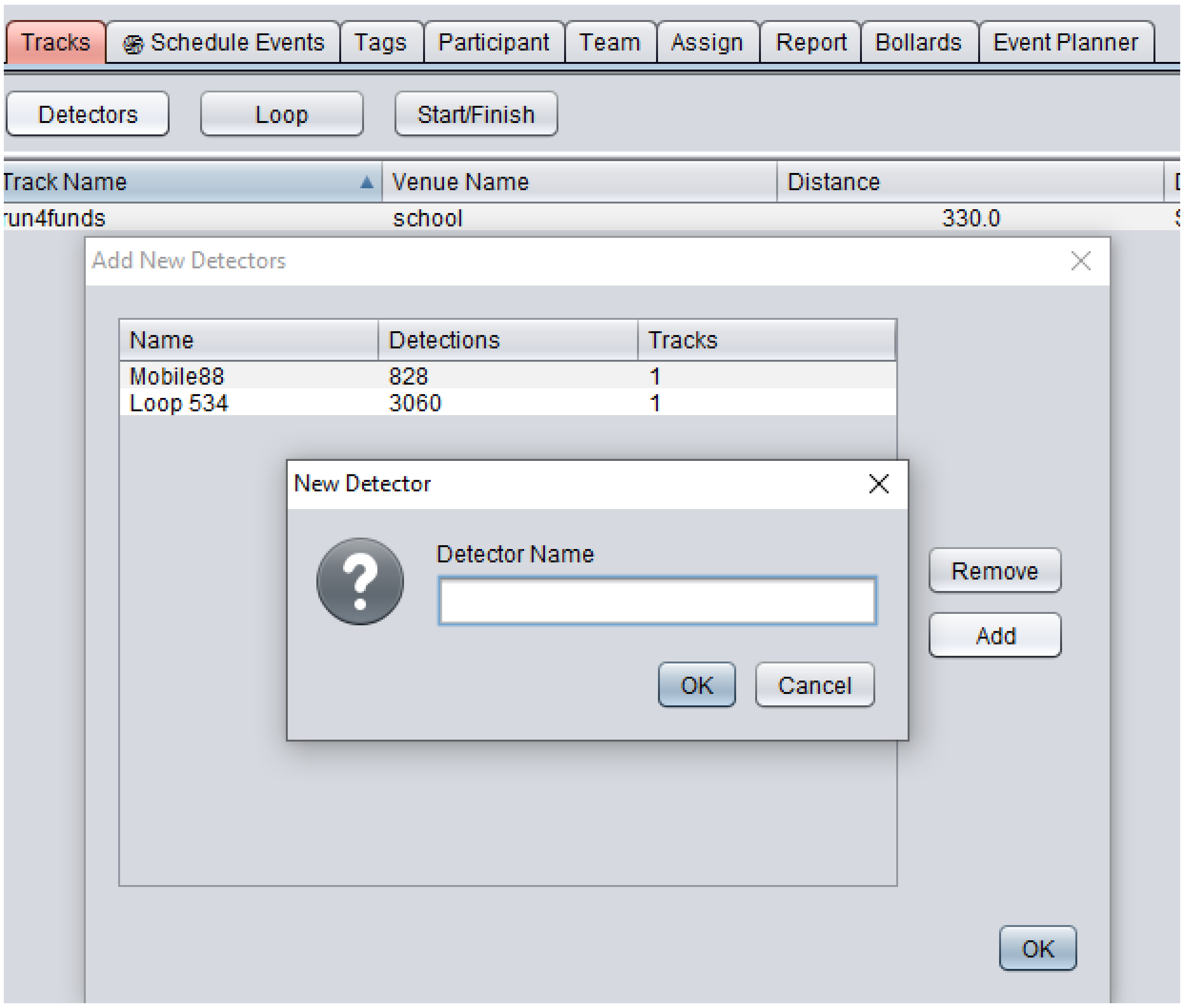
In Summary, there are two main screens for operating the Orbiter.   First is the “Home Page” that shows the tabs moving from left to right. Second, is the Green “Start and View” Events screen. Understanding these two screens allows set up and control of your event. To set up an event just move from left to right on the tabs filing out the fields, then go to the “Start and View” events, and start your event.

No worries if you fill out the event wrongly because you can always recover and redo your event from the original data found inside your readers. There is nothing you can do wrong with the software because if you (1) Turn your readers “on” (2) Know who you gave the tags to, (3) Make the reader beep when people pass, you are 100% and your race is safe.

With Orbiter the software is secondary. The readers are your race. The original tag detect data is stored time stamped inside. As long as you record the start time of your event, you don’t even need a live connection to the readers for 100% success of your race.

Now for the detail of each screen shot moving from left to right on the home page.

C.1.1 Tracks Tab Add Detectors

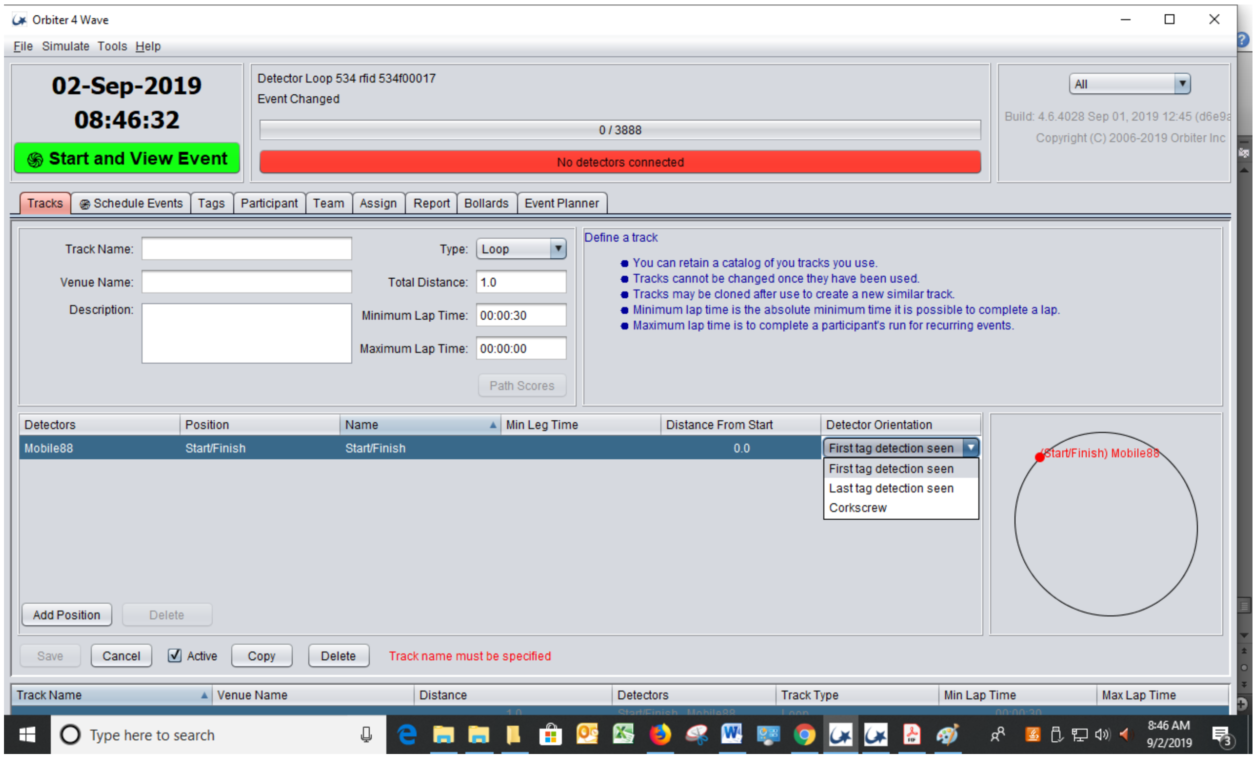


Track Name	Venue Name	Distance
run4funds	school	330.0

Name	Detections	Tracks
Mobile88	828	1
Loop 534	3060	1

In order for Orbiter4 to work, detector(s) need to be assigned first.   This is done by connecting a live connection to your detector such that under "Bollard Tab" there is a long green line showing a live connection. The software will then auto connect and self-enroll the detector such as that shown above. Or you may manually assign a detector under Detector Name. The lay out must be capital "M" with the detector number with no "space" between Mobile and the number. The detector number may be found on the Orbiter detectors. Mobile88 for example is short for IP address 172.25.0.88.   Only use the MobileXX nomenclature when assigning detectors.

C.1.2 Loop and Start Finish Tracks


Orbiter 4 Wave

File Simulate Tools Help

02-Sep-2019
08:46:32

Start and View Event

Detector Loop 534 rfid 534f00017
Event Changed

0 / 3888

No detectors connected

All

Build: 4.6.4028 Sep 01, 2019 12:45 (d6e9a
Copyright (C) 2006-2019 Orbiter Inc

Tracks | Schedule Events | Tags | Participant | Team | Assign | Report | Bollards | Event Planner

Track Name:
Venue Name:
Description:
Type: Loop
Total Distance: 1.0
Minimum Lap Time: 00:00:30
Maximum Lap Time: 00:00:00
Path Scores

Define a track
- You can retain a catalog of you tracks you use.
- Tracks cannot be changed once they have been used.
- Tracks may be cloned after use to create a new similar track.
- Minimum lap time is the absolute minimum time it is possible to complete a lap.
- Maximum lap time is to complete a participant's run for recurring events.

Detectors	Position	Name	Min Leg Time	Distance From Start	Detector Orientation
Mobile88	Start/Finish	Start/Finish		0.0	First tag detection seen

First tag detection seen
Last tag detection seen
Corkscrew

Start/Finish) Mobile88

Add Position | Delete

Save | Cancel | ☑ Active | Copy | Delete | Track name must be specified

Track Name	Venue Name	Distance	Detectors	Track Type	Min Lap Time	Max Lap Time

Type here to search

8:46 AM
9/2/2019

After assigning a detector the next choice is using a "Loop track" or a "Start Finish" Track. A loop track is where the same detectors are used for the start finish. This is true whether the start and finish are located geographically in different locations. For example, there can be a Triathlon where the Start detector is moved to the finish after the participants started. Or, there can be a 5K where the finish is a distance away from the start.

Tip: Be sure you can move quickly from the start to the finish before runners get to the finish. An advantage to Orbiter is it makes moving detectors a snap. Unlike mat systems that take time to move due to all the wiring can cabling.

A Start Finish race requires a detector be placed at the start and a different detector at the Finish. In this case, detectors cannot be moved and are permanently placed for the entire race. Examples of these types of races include hill climbs that use chip time individual starts. As a note, a way around having to have two detectors is using Interval starts. Interval starts may be started manually by the timer either at preset time intervals or with a push of the button for each participant. There is also a control to "pause" an interval start. Intervals are commonly used in Europe for cross country ski races.

Looking to the right of the Track GUI, is the "Min Lap" Time Setting. This is an important setting because it prevents participants from standing in front of a detector and being counted twice. For school lap counting, it prevents students from cheating and getting free laps. For 5Ks it allows walkers to chat in front of the reader before heading out for the race. Otherwise, you would have walkers winning a race as they linger at the start. It is always best practice to write down the winners bib number so any false tag detects can be deleted. Some ways this happens is when a volunteer brings no show participant pre-registered bibs to the finish to watch. These bibs then get detected after minimum lap time and create confusion as they appear to have won the race.

Next down on the screen is "Max Lap" time setting. This is used for unattended automated courses such as indoor and permanent outdoor tracks. It allows for results to be automatically erased so a new work out session can happen. For example, at a resort where there is an Orbiter unattended automated race course, Max Lap can be set so each day's workout is fresh for the participant to see. This eliminates clutter on the display screen from prior workouts.

Moving down the screen is how the detectors positions are set. It is important to know that by clicking on the detector name, more than one detector may be set to work in unison with another detector. This is done often at a start and finish such that a right and left detector work together as one. Adding and deleting detectors is easy by clicking right on top of the detector name. Not on the Detector column heading. Just add and remove detectors this way.

Tip about Position: By setting detectors to “Start/Finish”, the detector is assigned position “0”. This means the detector can be moved from the start to the finish. If all detectors are assigned position “Start/Finish” they can be moved along the race path on the fly. Thus picking up detections at each way point that is greater than minimum lap time. It is a slick way to use fewer detectors and economize by leap frogging them as participants travel on the race path. However, if a detection is missed the results will slide to the left. For example, for four detection points with one missed detection, then there will be three shown. In order to determine where the detection was missed, the timer must hover the cursor over the time shown in the real time event viewer and a listing of tag detects will be shown.

To simplify this, Orbiter4 allows detectors to be assigned a specific location under “Add Position”. This is found at the bottom of the Track GUI. Once a detector is assigned a “Position” it cannot be moved during the race. The detection is then pin pointed for the results and instant confirmation the detection is a “finish” detection is made. It’s best practice is to keep detectors operating at way points and not move them. However, the “Tip” above about setting all detectors to Start/Finish is a work around for economy.

“Min Leg Time” setting is used where the race is a start finish and Laps are set to “1” under Schedule events. This is the time it takes to get from one detector to another within the race. It is used for Triathlons where transition times are very short. As short as 10 seconds while min leg time for a run may be 15 minutes. Min Let Time is similar to Min Lap time as it keeps counting the tag for a reasonable period of time so the times are reported rationally.

Lastly, an important setting unique to Orbiter is “First Tag Seen”, “Last Tag Seen”, and “Corkscrew”. Since Orbiter readers are autonomous a real time connection to the server is not needed for positioning of tags.

Tip: When a beep is heard the location of the time is instant and may be seen immediately. This allows calibration of the reader positon on the spot. Calibrate the reader by moving it such the tag is detected on the line desired.

Last tag seen is useful for Bike races by pointing the detector at the riders to pre-energize the passive RFID for best responsiveness.

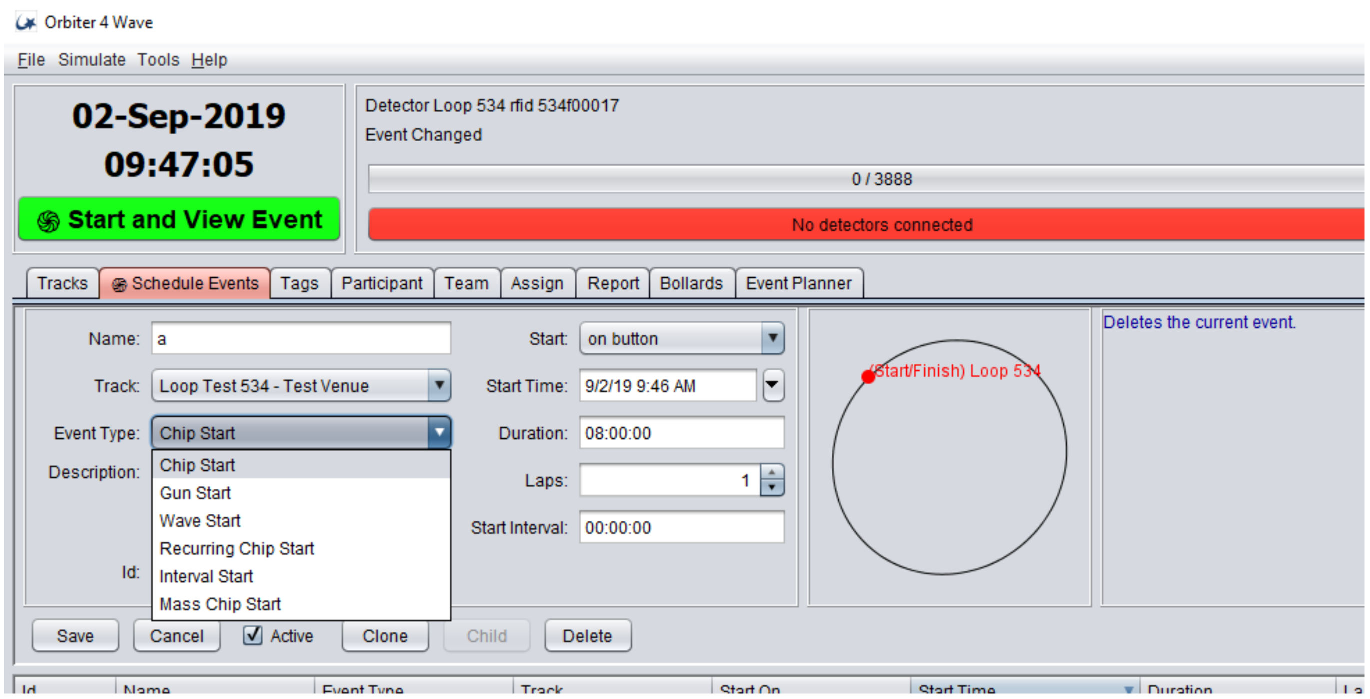
Corkscrew is used for two readers such that one is pointed toward the participant and another way. This allows reading tags placed on both front and back of a runner. In a group of 100 runners there are almost always those that put the tags on their backs or sides. Corkscrew solves reading these tags. Another method is to use the SLING reader that is highly mobile. It can be easily turned quickly so that all tags may be read both front and back.

Using the above methods means a separate “Time Machine” 10 key manual entry system is not needed. The rule with Orbiter is, “If you can see the tag, the detector will read the tag. If you cannot see the tag, then you must assume the reader cannot read the tag.” Either the Corkscrew needs to be used, or moving the SLING right or left as needed to read the tag.

A note on Event Type, “PATH”: Path events are unique to Orbiter and are fun. You may place multiple detectors and have races that go from one to another in random order. Hit all 10 detectors and then finish the race. Strategy may be used to win these unique races.

C.2.1 Schedule Event Tab

To set up an event, Click on “New Event” and you will then see this screen.


Orbiter 4 Wave

File Simulate Tools Help

02-Sep-2019
09:47:05

Start and View Event

Detector Loop 534 rfid 534f00017
Event Changed

0 / 3888

No detectors connected

Tracks | Schedule Events | Tags | Participant | Team | Assign | Report | Bollards | Event Planner

Name: a
Track: Loop Test 534 - Test Venue
Event Type: Chip Start
Description:
Id:

Chip Start
Gun Start
Wave Start
Recurring Chip Start
Interval Start
Mass Chip Start

Start: on button
Start Time: 9/2/19 9:46 AM
Duration: 08:00:00
Laps: 1
Start Interval: 00:00:00

Deletes the current event.

Save | Cancel | Active | Clone | Child | Delete

Id | Name | Event Type | Track | Start On | Start Time | Duration

Fill out a name for the race. You can change any of the settings at any time, so do not worry about filling the forms wrongly. You can come back and correct errors or change your mind later. Next choose your track. The reason for moving from left to right on the tabs is that what is filled out and built on as we progress to ready and Start the race. Track is an example. You must have filled out the “Track Tab” form previously in order to choose a track.

The “Event Type” allows you a wide choice of events. Here is an explanation for each:

Chip Start provides each participant an individual start time when their RFID transponder passes the detector and the detector “beeps”. Under the Green “Start and View” events button, choosing the event and then clicking the “Start” button readies the Orbiter for a Chip Start. The participants start time is when he passes the Orbiter reader, not when the Start button is clicked. Clicking the Start button only readies the system for a chip start. Results on the View Events page are displayed as “Elapsed Time”. This is the time from the first RFID chip detection to the tag detection to the reader. For Chip Start this is the Finish Time too. “Most Recent Tag Detect” is the time from clicking the start button and the most recent tag detect. It is also called Gun Time.

When “Gun Start” is used, the most recent tag detect and Elapsed Time are the same. Start on Gun Start happens when the start button is clicked. This provides the same start time to all Participants.

Wave Start is set for groups of people to start in Waves. In order for this to work, the All-In-One excel must be filled out with assigned wave numbers. Or, Dynamic Waves may be made by self-enrollment of the participants as they approach the start reader. Starting a Wave is done in the Green Start and View Events real time Event Viewer page. Wave starts are on the left side of the page under Wave, and then Start Wave. To start the Wave, click “Start Wave”. It is good to practice starting waves by “Cloning” a practice event prior to your race. Practice false starts and then restarting a wave quickly.

Reoccurring Chip Start is used for indoor tracks and resort automated trail runs where participants repeat the same event daily or within a set time interval. Using this type of Start allows the prior work out not to show on the display. The display may be “cleared” of a prior work out so only the current work out session is shown. The time interval to clear the display is called “Max Lap Time.”

Interval Start is where a present time is set to start individuals in a race. This is done by setting the “Start Interval” time. The start interval then automatically starts each racer on a cadence, such as 15 second intervals if the start time is set at such. An interval start number must be assigned to each participant in the Excel All-In-One. This is used for cross country ski races in Europe and other elite sports. Interval Start has a pause and resume button too.

Where interval starts are desired such that each person starts with a push of the start button, Wave Starts may be used. Each person in the all-in-one is assigned an individual wave number. This is used for Indoor Pool Triathlons where the participants start one at a time at one corner of the pool, and then swim (snake) their way back and forth down each lane of the pool, to the exit of the pool on the far end. These types of races are common for YMCA’s and military bases where lake Triathlon races are not possible.

Interval Starts and Wave Starts are a good way to reduce the need for extra Orbiter readers by eliminating the need for a reader at the start.

“Mass Chip Starts” pegs the start time to the individual as if it is a gun start. If there is a missed tag detect the click of the “Start” button is used. To reduce the chance of missed start times use two Orbiter SPIRE detectors to reduce shadowing at the Start. All passive RFID have as much as a 5% missed tag detects at the start. Mats are the same as Orbiter in this respect with missed detects often higher because of the antennas placed on the ground unlike Orbiter in the air. UHF radio frequency works better with air to air communication using side antennas.

“Start” may be set to “On Button” or Start “As Scheduled” drop down. Start on button setting is used to ready the event for Chip Start. The actual participant time starts as the RFID tag passes the Start Reader. Start on Button is also used for Gun Starts. The majority of events are using Gun Starts as this the running rules. Otherwise, it is confusing to spectators as to who won a race because the first person across the finish line may not be the winner. However, Chip Starts are preferred for individuals that are at the back of the pack. Thus, having results both as Chip Start and Gun Start is good. This is done by setting to Chip Start and then displaying both Most Recent Tag Detects (Gun Start) and Chip time (Elapsed Time).

“Start as Scheduled” is typically used for school PE classes when the “Event Viewer” tab is used.

When “Start Time” is set, the computer will be “Ready” to start one hour before or one hour after the set time. If a mistake is made, then on the Green Start and View events on the “Event Viewer” the time will show how many hours until the start will happen. To by-pass this, just push the “Start” button and your event will start without the need to reset to “Ready”.

“Duration” is used to automatically stop an event. It is used to turn off events if the timer forgets to do so. If an event is turned off inadvertently, use the “Resume” button found in the Green Start and View Events area of the program. It is located right under the Start button that started your event. Duration is also used for setting up physical education class sessions under the “Event Viewer”.

“Laps” is a most important setting. Setting Laps = “1” is used on a start finish race like a 5K. It shows a participant’s time from the start of the event to the end of the event, even if the person comes back to the finish line root their friends on. Setting to one will ignore further tag detects and post accurate times by ignoring additional detections. If Laps is set to 4 for a mile run, then only the mile run is timed. If laps is set to 6, then a military or police fitness test appropriate laps are counted. In event laps are set incorrectly, then setting laps correctly after the fact and pressing “Reprocessing” fixes the problem. Reprocessing is found in the Green Start and View events area.

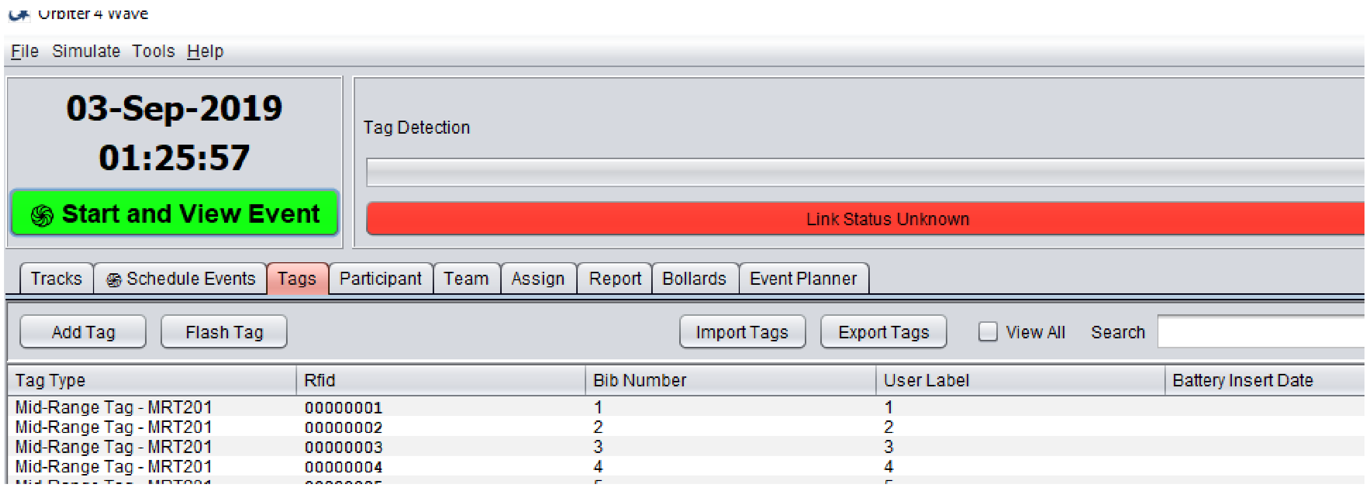
“Start Interval” is the time set to automatically start participants in Intervals such as every 15 seconds or every 30 seconds. Start Interval only works with the event is set to Interval Start type.

“Clone” is to copy an event. It may be used for repeat events such as Fitness Testing. Thus with just a few key strokes repeat back to back events may be started. “Clone” is also a second way other than “Resume” to start a closed event.

“Child” is very powerful way of creating subs events to a Master Event. By creating children to a Master you can report the sum of all events quickly to the Master event. For example, this can be used to create a vertical ski program where each chair lift is its own event, yet the sum of all chairlifts are counted for the overall event. If you are part of resort network of automated trail runs, you can link each trail run into a total trail run. Or, if you are a school, individual schools to a district wide event.

“Delete” is where you delete an event. In order to delete tracks you must first delete the event and then the track. It is always good to back up your database prior to deleting events. This way you can recover your data if you make a mistake. Backup database is found in the upper left of the Orbiter4 GUI under “File”.

C.3.1 Tags



Tag Type	Rfid	Bib Number	User Label	Battery Insert Date
Mid-Range Tag - MRT201	00000001	1	1	
Mid-Range Tag - MRT201	00000002	2	2	
Mid-Range Tag - MRT201	00000003	3	3	
Mid-Range Tag - MRT201	00000004	4	4	

Tags tab is where RFID transponders are imported into the computer. With Orbiter there are typically three ways to do to gain the same result. Tags are no exception. The first and easiest way to Import tags is to “Export” a Tags Excel Template and fill it out and import your tags. When you purchase inlays from Orbiter, you are sent a Tags file to import. When you use an “Enrollment Machine” an excel tags file is automatically built for you. You import this file. This is the easiest way to import hundreds if not thousands of tags.

A second way to is “Add” a tag individually. To do this select the Tag Type.

Standard Tag is a High Frequency 13.56 mHz Tag such as Near Field passive tag. It is used for concert and sporting event access.

Medium Tag is the Standard 800 – 900 mhz UHF passive tag used by most road running race events around the world.

Long Range Tag is an active tag with a battery operating at 2.45 GHz. This tag is unique as it is always “on” and does not blink or power up as other active tags do. This tag is used for motor sports and military applications. There is no need to recharge the battery after every use as is normal in other active systems. Battery life is for Long Range tags to remind you when the tag needs to be replaced. All active tags purchased at the same time, will be replaced at the same date. This is 5 years after purchase. With Orbiter active tags, you will not have tags fall out as some are used more than others. This ensures good results at your races as there is no worry of some tags failing and other not due to lower battery power.

Bib Number and User Label are normally the same. However, User label is most important as this number is what binds the bib (tag) to the runner. The system will not work without a bib number.

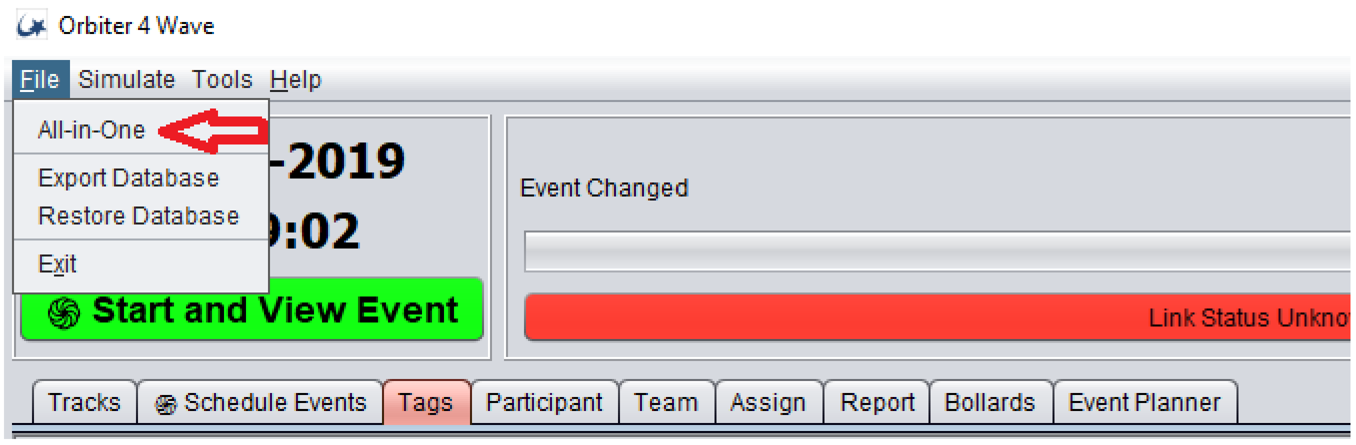
RFID number is the number found inside the integrated circuit. It is an 8 Hex number. If you want to see this number, turn on the Orbiter reader and connect a live connection to your computer gaining a green line on “Bollard” tab. Then scan the tag and you will see the RFID number inside the chip.

You can edit RFID tags by double clicking on the long white line with the tag shown. When you do this you will see you can designate a tag a “Start Tag”. This them makes the tag when shown to the reader, start your event. It is an optional way to start an event, without the need for a lap top being present.

Tip: Tags are the “Key” in the database. All tags used in the event must be loaded in advance. No new RFID tag numbers may be added after the event has started unless it is done automatically under the Green “Start and View Events” button, “Start Events”. Click the “Add if Missing” box.

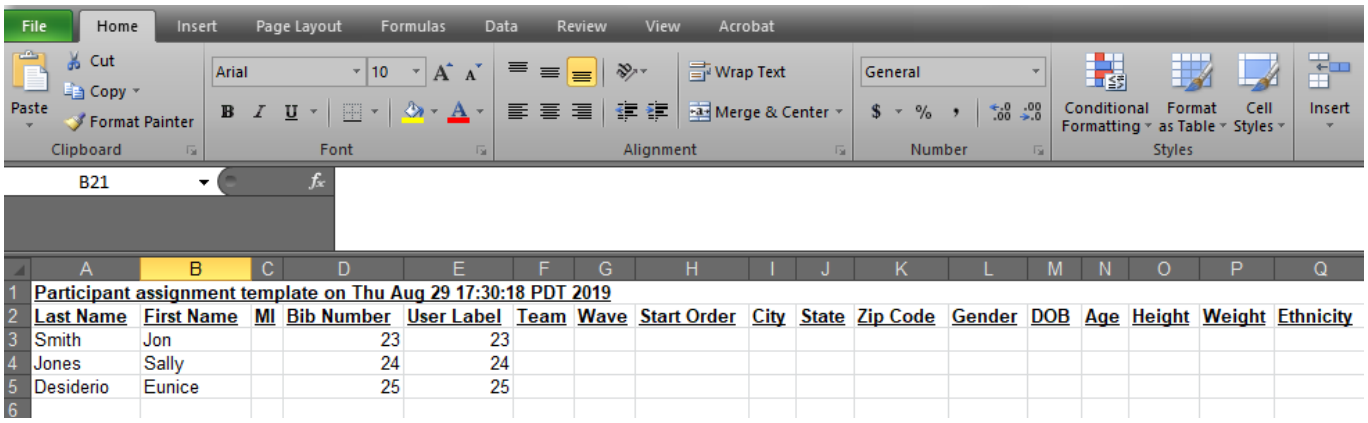
D.1 Loading your Participants and Associating tags.

The three tabs that must be filled out for the building blocks to enroll participants is Tracks tab, Schedule Events Tab, and Tags Tab. Once these are done you can move onto loading your participants and association your RFID tags to them. Then lastly, start your event and review results either live with the real time viewer or in logging mode by using reports.



There are two ways to do this. First is the quick easy way. Go to the upper left of the Orbiter GUI, and select “File”, “All-In-One”. An Excel Spread sheet will open, cut and paste into it, save it to a place on your computer you can find it, and then import.

Once the All-In-One is produced be sure to fill out four columns at a minimum. Last Name, First Name, Bib Number and User Label. Bib Number and User label are normally the same number, where User Label is the most important of all. User Label is where the bib is associated with the name. If there is no User Label results will not show up until you input a user label. Middle Initial is a single Initial and not a full name.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Participant assignment template on Thu Aug 29 17:30:18 PDT 2019																
2	Last Name	First Name	MI	Bib Number	User Label	Team	Wave	Start Order	City	State	Zip Code	Gender	DOB	Age	Height	Weight	Ethnicity
3	Smith	Jon		23	23												
4	Jones	Sally		24	24												
5	Desiderio	Eunice		25	25												
6																	

For the Second method of putting for loading participants is to continue moving from tab to tab left to right. Fill out the series of Tab to Participants, Team, and Assign. This method is for individual edits and is tedious unless there are a few to check. Individual edits are a useful tool and will be explained in detail later same as the Assign and Bollard Tabs.

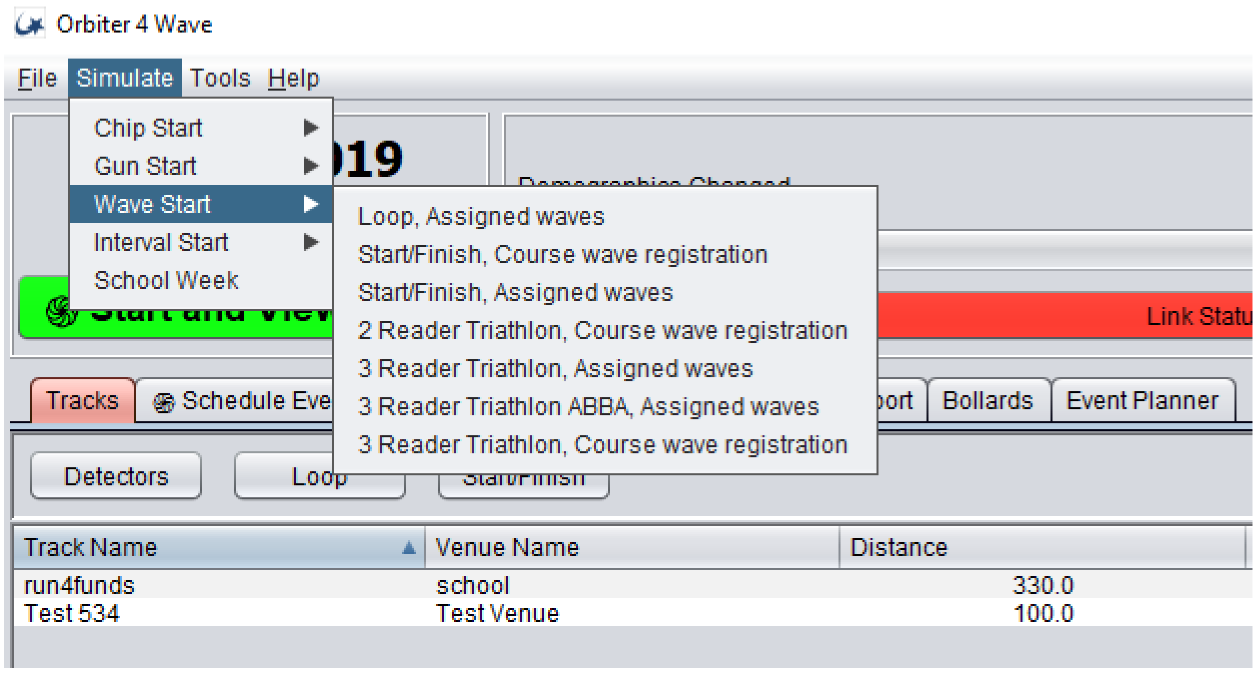
The "Assign" tab is where you can double check that all your participants are loaded and in the event. The "Bollard" tab is where you can check to see you have a live connection from your Readers to your Computer.

Reports are most important and a future section will explain these later too.

Event Planner is for Schools and very unique as it allows for tags to be traded between participants and automate turning off and on many events during a semester. Given the average PE teacher has 5 to 6 classes a day and each is an event, the event planner saves the teacher time.

On the drop down notice "Export Database" which is the same as Backup Database. "Restore Database" is the same as "Import Database". It is always good to back up your database before making major changes to configurations of repeat events.

D.3 The SIMULATE: Examples of Configuration of Various Events.


Orbiter 4 Wave
File
Simulate
Tools
Help
Chip Start
Gun Start
Wave Start
Interval Start
School Week
Loop, Assigned waves
Start/Finish, Course wave registration
Start/Finish, Assigned waves
2 Reader Triathlon, Course wave registration
3 Reader Triathlon, Assigned waves
3 Reader Triathlon ABBA, Assigned waves
3 Reader Triathlon, Course wave registration
Link Statu
Tracks
Schedule Eve
Bollards
Event Planner
Detectors
Loop

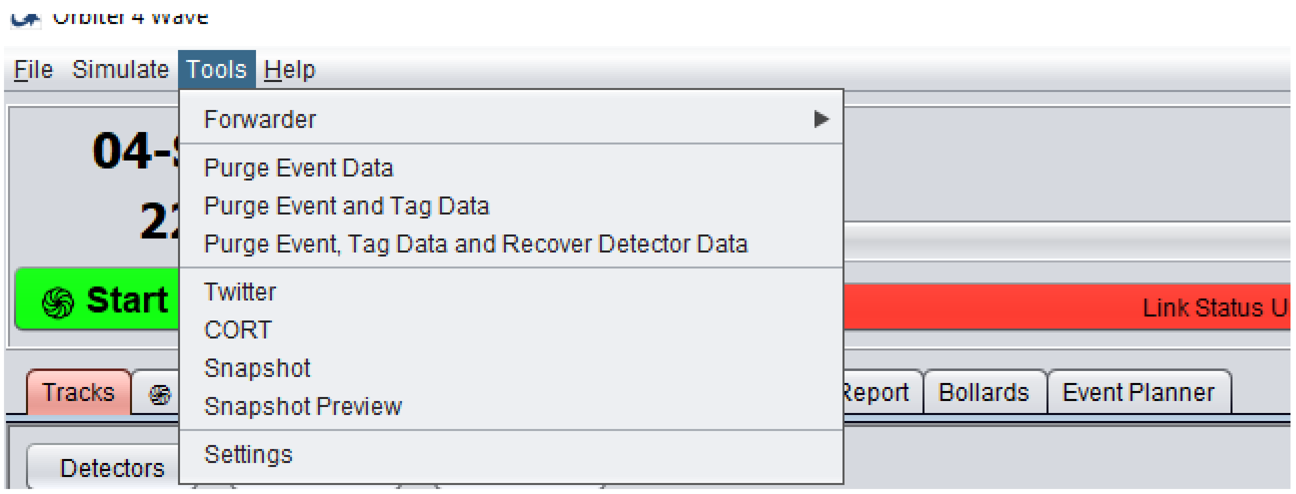
Track Name	Venue Name	Distance
run4funds	school	330.0
Test 534	Test Venue	100.0

The Simulator is a good way to check the configurations of various events. By running a simulator you can look at the details of Tracks, Schedule events note the set up of each field to double check if your event is set correctly.

A good example is to run the simulator for a 3 Reader Triathlon with ABBA. Where ABBA is the standard triathlon transition box between (Transition 1) the Swim and Bike; AND (Transition 2) Bike and Run Portion of the race. "A" being the entry to the box and "B" the exit on Transition 1. The opposite being "B" the entry and "A" for the Bike to Run portion of the Race.

Practice navigating the Orbiter4 software program while an event is running. First, go to the Green "Start View Events" button, choose your event, and then open up the "Event Viewer". You will see live results as the event progresses. Then go back to the Orbiter4 home page. This is where the tabs move from left to right. Click on the "Bollard" tab and you will see what a live connection looks like from the database. Bollard condition monitors each RFID reader when there is a live connection. Each green line on Bollard control represents one detector. Remember, that Orbiter can also operate in logging mode. Logging mode means no live connection and down load of data happens after the event. Only then will a green line show a live connection. This allows the system to work both in logging and real time mode.

D.4 Tools



The tools area provides control for "Forwarding" reader tag detects in "slave" mode. This is how traditional RFID readers operate. Raw tag time stamped tag detect data is sent to the server. Data may be quickly purged to "clean" the database. Another method to completely clean the database is to go to C: drive, Program Files, Orbiter, Orbiter4, and delete the "**db**" folder.

Results may be sent via "Twitter".

"CORT" is where connectivity may be made to individual timers for Registration and credit card processing companies.

"Snapshot" is for connecting the RFID reader to a camera such that pictures are automatically taken when a participant passes. The Snapshot preview is used to sight in the picture frame and preview the picture before a race is started.

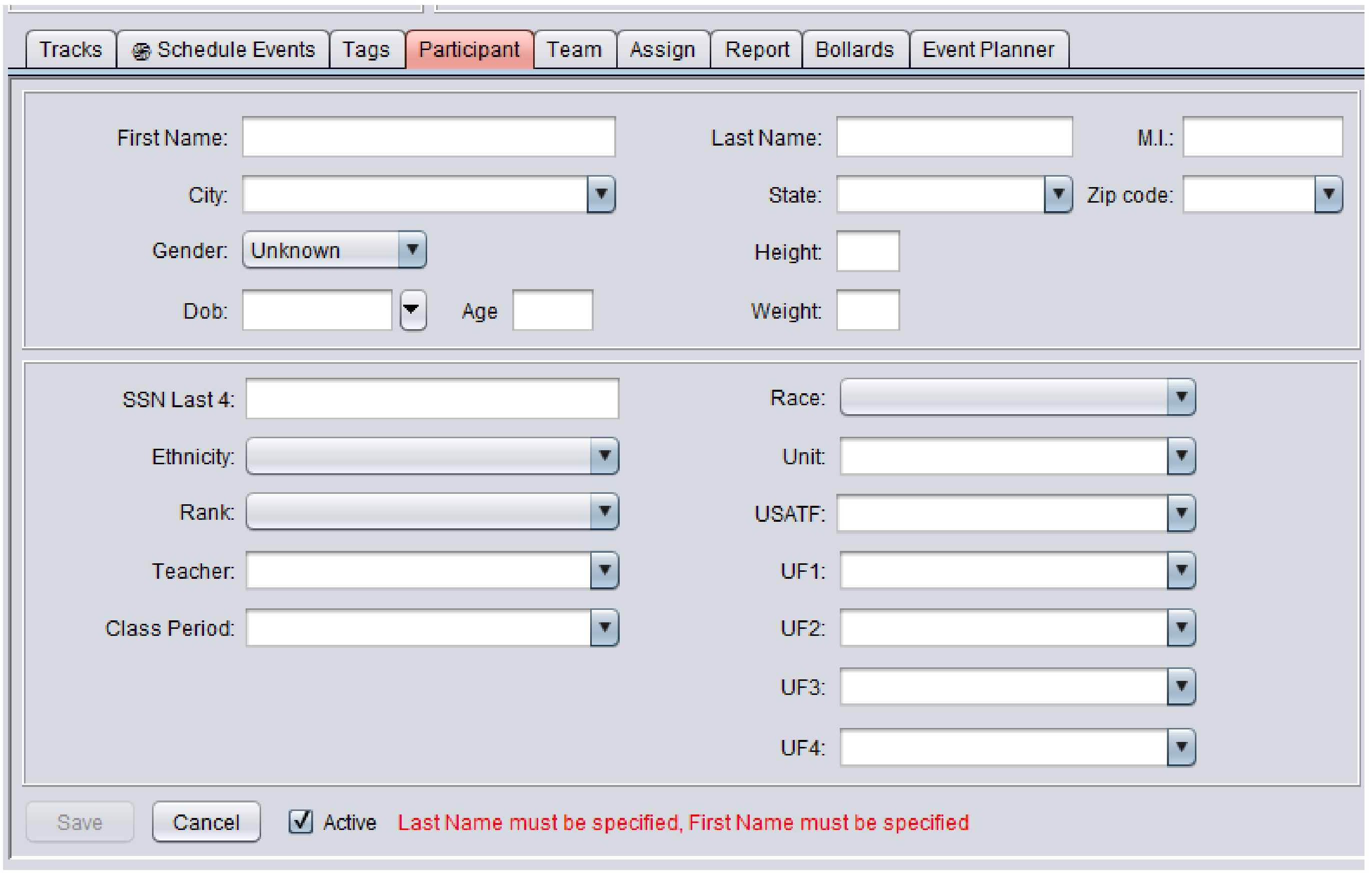
"Setting" has various controls for the software such as setting type set and pixel area.

D.4 Back to working Left to Right on the Tabs

We interrupted the explanation of each Tab from moving left to right, because when it comes to filling out the Participants in an event and associating them with their tags, it is far easier to use the "File" "All-In-One" short cut than continuing moving one by one from Participant to Assign tabs.   However, for individual edits the tabs method is easier.

We now should continue with the explanation for individual edits of participants.

D. 5 : Participant Tab



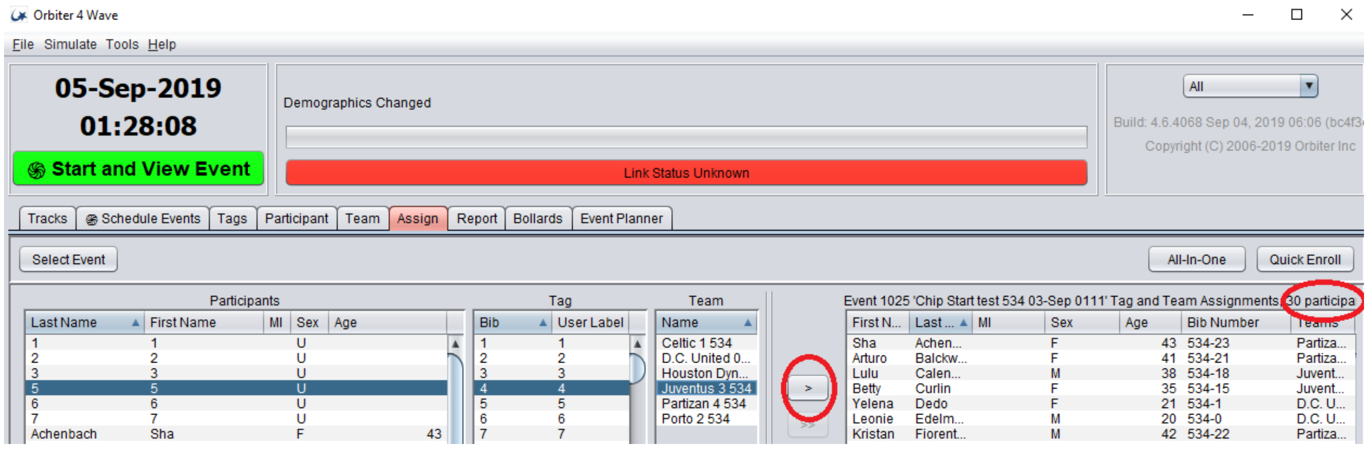
First Name:		Last Name:		M.I.:	
City:		State:		Zip code:	
Gender:	Unknown	Height:			
Dob:	Age	Weight:			
SSN Last 4:		Race:			
Ethnicity:		Unit:			
Rank:		USATF:			
Teacher:		UF1:			
Class Period:		UF2:			
		UF3:			
		UF4:			

[✓] Active   Last Name must be specified, First Name must be specified

When filling out the fields only two are required, Last Name and First Name. USATF is a number used for qualifying for the Boston Marathon. UF fields are user defined.   You can put whatever you want in these.   For example, you can designate UF1 for Clysdales.

The "Team" tab is where you put team categories. For schools "Team" is often the same as "Teacher".

D. 5 : Participant Tab



05-Sep-2019
01:28:08

Demographics Changed

Link Status Unknown

Build: 4.6.4068 Sep 04, 2019 06:06 (bc4f3

Copyright (C) 2006-2019 Orbiter Inc

Participants

Last Name	First Name	MI	Sex	Age
1	1		U	
2	2		U	
3	3		U	
5	5		U	
6	6		U	
7	7		U	
Achenbach	Sha		F	43

Tag

Bib	User Label
1	1
2	2
3	3
4	4
5	5
6	6
7	7

Team

Name
Celtic 1 534
D.C. United 0...
Houston Dyn...
Juventus 3 534
Partizan 4 534
Porto 2 534

Event 1025 'Chip Start test 534 03-Sep 0111' Tag and Team Assignments 30 participa

First N...	Last ...	MI	Sex	Age	Bib Number	Teams
Sha	Achen...		F	43	534-23	Partiza...
Arturo	Balckw...		F	41	534-21	Partiza...
Lulu	Calen...		M	38	534-18	Juvent...
Betty	Curlin		F	35	534-15	Juvent...
Yelena	Dedo		F	21	534-1	D.C. U...
Leonie	Edelm...		M	20	534-0	D.C. U...
Kristan	Fiorent...		M	42	534-22	Partiza...

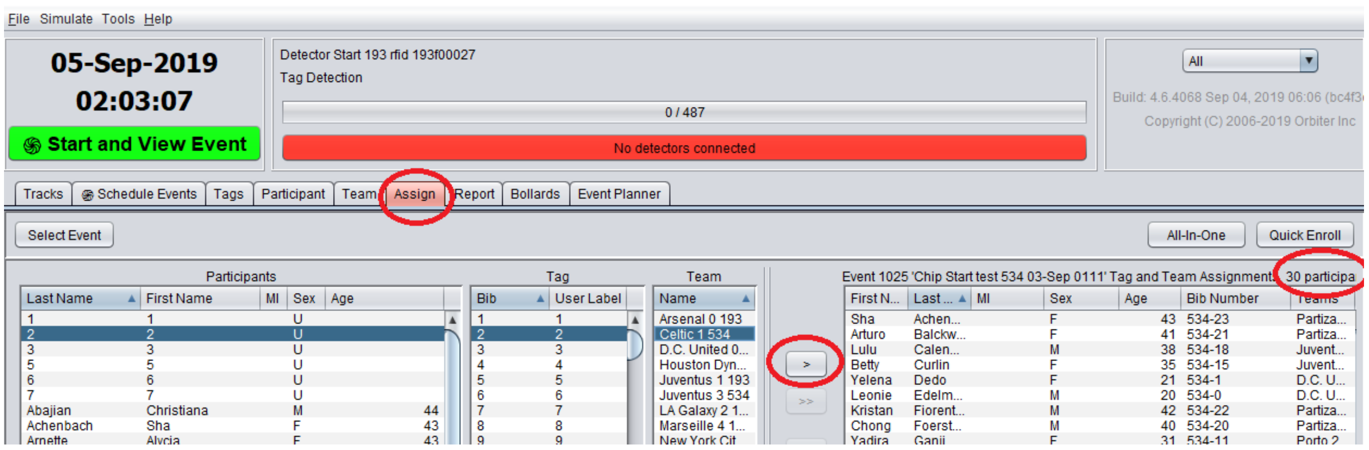
The Participant tab is an important tab because it is where you double check to see if everyone is in the race. The screen has a left and right side separated by “>” arrows. The left side is where Participants, Tags and Teams are assigned. The right side is where the participants in the event are located. To put a new person in an event, “select event”. Then then click on the left side of the pate area and highlight Participant, Tag and Team. They will highlight in blue. Then click the Arrow “>”found in the middle of the page, and the participant with their associated tag is put in the event. To take a person out, do the reverse “<”.

Most important is to check the number of people in your event. On the upper right of the left screen you will see the number. Thus if you have 100 people pre-registered, and then add 50 people day of the event, this number should be 150 participants. It is an important way even with using the “All-In-One” to be sure you loaded your event correctly. If not, just reload your event. You can add people after the event starts or even after the event. Just click the “Reprocess” button found in the Green Event Viewer to make the data refresh to see the updated race results that include the new people.

On this page, is “All-In-One” for creating the important all in one spreadsheet template. This is a second place you can create an All-In-One. The other location is on the upper left of the home page GUI under “File”.

Quick enroll is used when the reader is connected to the computer and a tag is flashed at the reader.

E. 1 : Assign Tab



05-Sep-2019
02:03:07

Detector Start 193 rfid 193f00027
Tag Detection

0 / 487

No detectors connected

Build: 4.6.4068 Sep 04, 2019 06:06 (bc4f3

Copyright (C) 2006-2019 Orbiter Inc

Participants

Last Name	First Name	MI	Sex	Age
1	1		U	
2	2		U	
3	3		U	
5	5		U	
6	6		U	
7	7		U	
Abajian	Christiana		M	44
Achenbach	Sha		F	43
Arnette	Alycia		F	43

Tag

Bib	User Label
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9

Team

Name
Arsenal 0 193
Celtic 1 534
D.C. United 0...
Houston Dyn...
Juventus 1 193
Juventus 3 534
LA Galaxy 2 1...
Marseille 4 1...
New York Cit

Event 1025 'Chip Start test 534 03-Sep 0111' Tag and Team Assignment 30 participa

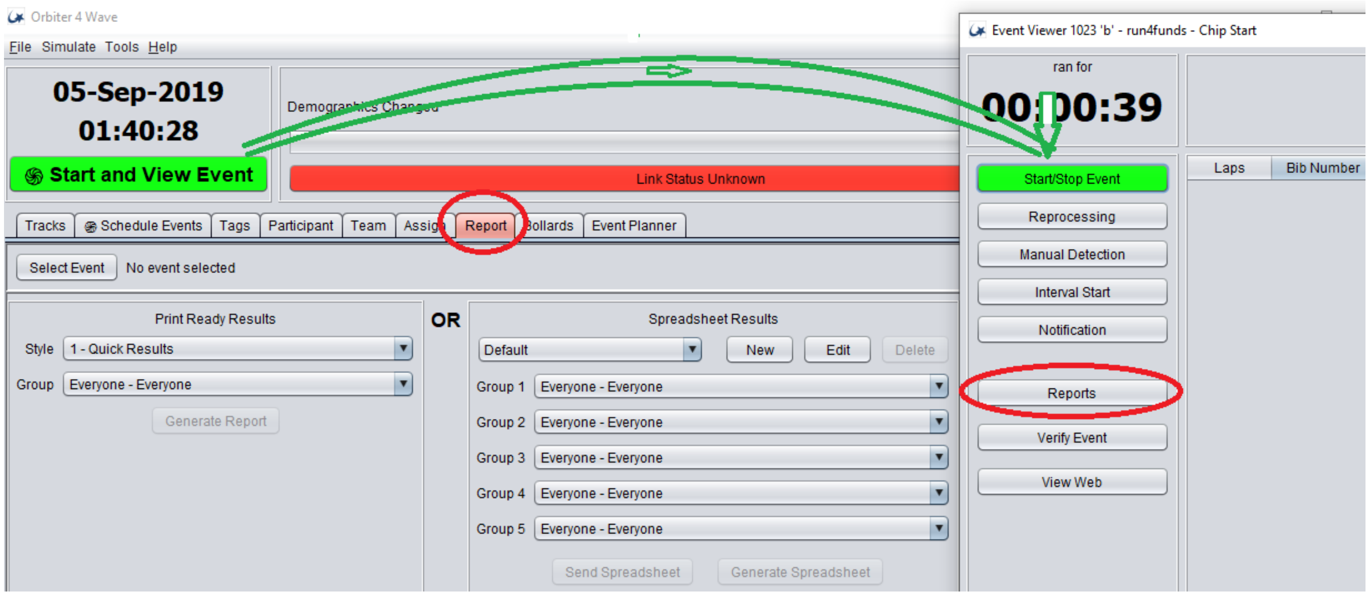
First N...	Last ...	MI	Sex	Age	Bib Number	Teams
Sha	Achen...		F	43	534-23	Partiza...
Arturo	Balckw...		F	41	534-21	Partiza...
Lulu	Calen...		M	38	534-18	Juvent...
Betty	Curlin		F	35	534-15	Juvent...
Yelena	Dedo		F	21	534-1	D.C. U...
Leonie	Edelm...		M	20	534-0	D.C. U...
Kristan	Fiorent...		M	42	534-22	Partiza...
Chong	Foerst...		M	40	534-20	Partiza...
Yadira	Ganji		F	31	534-11	Porto 2

First “Select Event” and then this is where Participants, Tags, and Teams (Left side of Screen) are assigned to Events (Right Side of the Screen) by using the “>” key. Most importantly it is where the number of people placed into an event may be double checked. Here is shown 30 people. The number one reason a person’s results do not show up is because they were never put into the event. To correct this problem, just find the person and bib, and put them in the race. Click “Reprocessing” on the Green Event Viewer page, and presto their results will show up.

You can sort people by clicking on the column header such as Last Name shown above. You can edit people names on the fly by double clicking on the name directly, and then editing it. This is good for correcting misspellings or correcting switched bibs between people. Also, switch the names by editing the name field, and never doing it by editing the Bib field.

E. 1 : Report Tab

The Report Tab is where custom reports are created. This is found on the home page. You will note that a second location for reports is on the Green Event Viewer where you start a race.

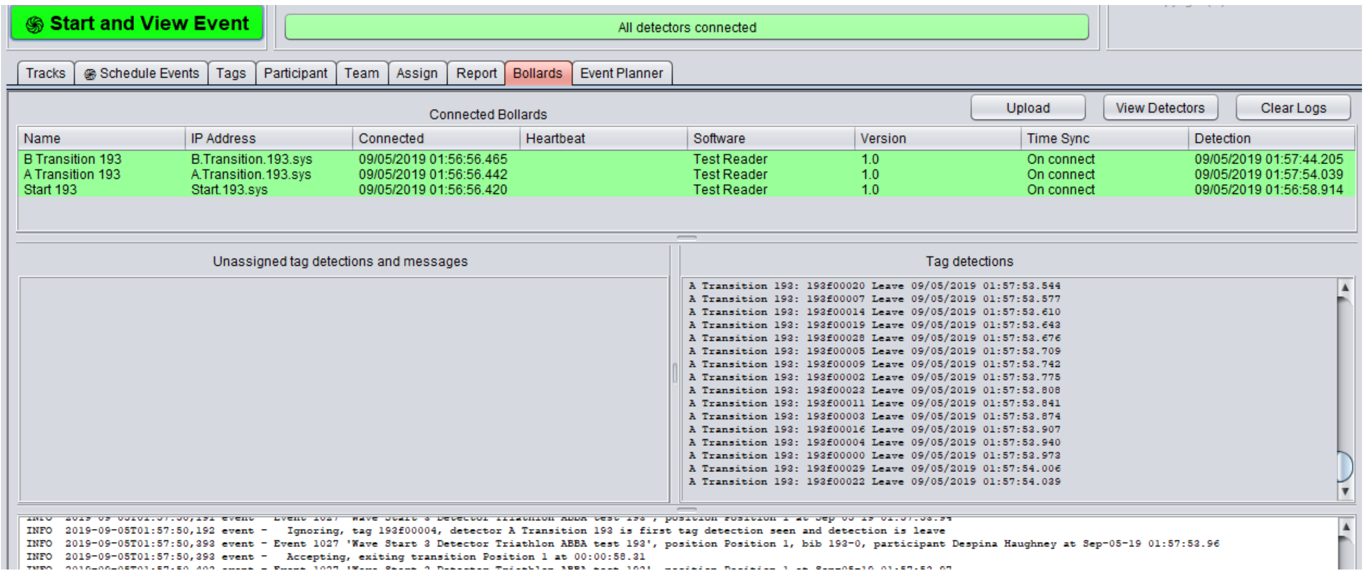


Reports will be covered in more detail in another section. However, here is an overview. Under the Report tab you will see two sides with “OR” separating them. On the left is Print Ready Results. These are .pdf results that are pre-made and standard. On the Right Side is where you make custom Print Ready Results. To do so, just click “New” type in the name and another view is shown listing all the column options for the report. In the screen area to the right are the column listing that will print. The first column on the page will be the top column listed. To remove column headers just click “<” and move them to the left, off the report. To put columns in the report click “>” to put them in the report. Then Save and your report will then appear as a Print Ready Option over on the “Reports” page on the Event Viewer.

For even more sophisticated reports Orbiter works with Jasper Reports. This is a World Class robust and professional report writer.   Jasper may be used to create stadium tall LED display output, television quality commercial output, and mobile phone applications like anything you have ever seen on your phone, at the movies, or TV.

To summarize, the areas to find reports are (1) the home page “Report Tab”, (2) the “Event Viewer” under Reports, (3) Real time results “Even Viewer” where live results are shown.

E. 1 : Bollard Tab: Live Reader Condition Monitoring.



Connected Bollards

Name	IP Address	Connected	Heartbeat	Software	Version	Time Sync	Detection
B Transition 193	B.Transition.193.sys	09/05/2019 01:56:56.465		Test Reader	1.0	On connect	09/05/2019 01:57:44.205
A Transition 193	A.Transition.193.sys	09/05/2019 01:56:56.442		Test Reader	1.0	On connect	09/05/2019 01:57:54.039
Start 193	Start.193.sys	09/05/2019 01:56:56.420		Test Reader	1.0	On connect	09/05/2019 01:56:58.914

Unassigned tag detections and messages

Tag detections

```
A Transition 193: 193f00020 Leave 09/05/2019 01:57:53.544
A Transition 193: 193f00007 Leave 09/05/2019 01:57:53.577
A Transition 193: 193f00014 Leave 09/05/2019 01:57:53.610
A Transition 193: 193f00019 Leave 09/05/2019 01:57:53.643
A Transition 193: 193f00028 Leave 09/05/2019 01:57:53.676
A Transition 193: 193f00005 Leave 09/05/2019 01:57:53.709
A Transition 193: 193f00009 Leave 09/05/2019 01:57:53.742
A Transition 193: 193f00002 Leave 09/05/2019 01:57:53.775
A Transition 193: 193f00023 Leave 09/05/2019 01:57:53.808
A Transition 193: 193f00011 Leave 09/05/2019 01:57:53.841
A Transition 193: 193f00003 Leave 09/05/2019 01:57:53.874
A Transition 193: 193f00016 Leave 09/05/2019 01:57:53.907
A Transition 193: 193f00004 Leave 09/05/2019 01:57:53.940
A Transition 193: 193f00000 Leave 09/05/2019 01:57:53.973
A Transition 193: 193f00029 Leave 09/05/2019 01:57:54.006
A Transition 193: 193f00022 Leave 09/05/2019 01:57:54.039
```

```
INFO  2019-09-05T01:57:50,192 event -   Ignoring, tag 193f00004, detector A Transition 193 is first tag detection seen and detection is leave
INFO  2019-09-05T01:57:50,393 event - Event 1027 'Wave Start 3 Detector Triathlon ABBA test 193', position Position 1, bib 193-0, participant Despina Haughney at Sep-05-19 01:57:53.96
INFO  2019-09-05T01:57:50,393 event -   Accepting, exiting transition Position 1 at 00:00:58.31
```

The bollard tab is not needed to run a race; however, it is a nice tool to let you know what is happening to all your readers in real time. Above you see three green lines. This shows three readers operating simultaneously. A green line means a live connection. To connect to a reader just turn the reader and the computer on. Be sure you have a live Ethernet, Wi-F-, Cellular, or VHF radio connection. Remember, you do not need to have an active connection to time a race. This is because Orbiter also works in logging mode and down load of the data can happen anytime including after the race. In logging mode there are no green lines and the screen is blank. This is normal.

In the screen above where there is a live connection to three readers for a Triathlon, you will notice two areas. “Unassigned Tag Detects” and “Tag Detections”. In the unassigned tag detections heart beat monitors will let you know if the readers are operating normally, also if any tags are seen that are not part of the race. Or, have not been assigned to the race. If you want to include these tags in to the race, go to the Green Start View Event page, click on the event, and then under the Start button, you will find “Create if Missing”. Check the box and the unassigned tags will automatically be entered into the race.

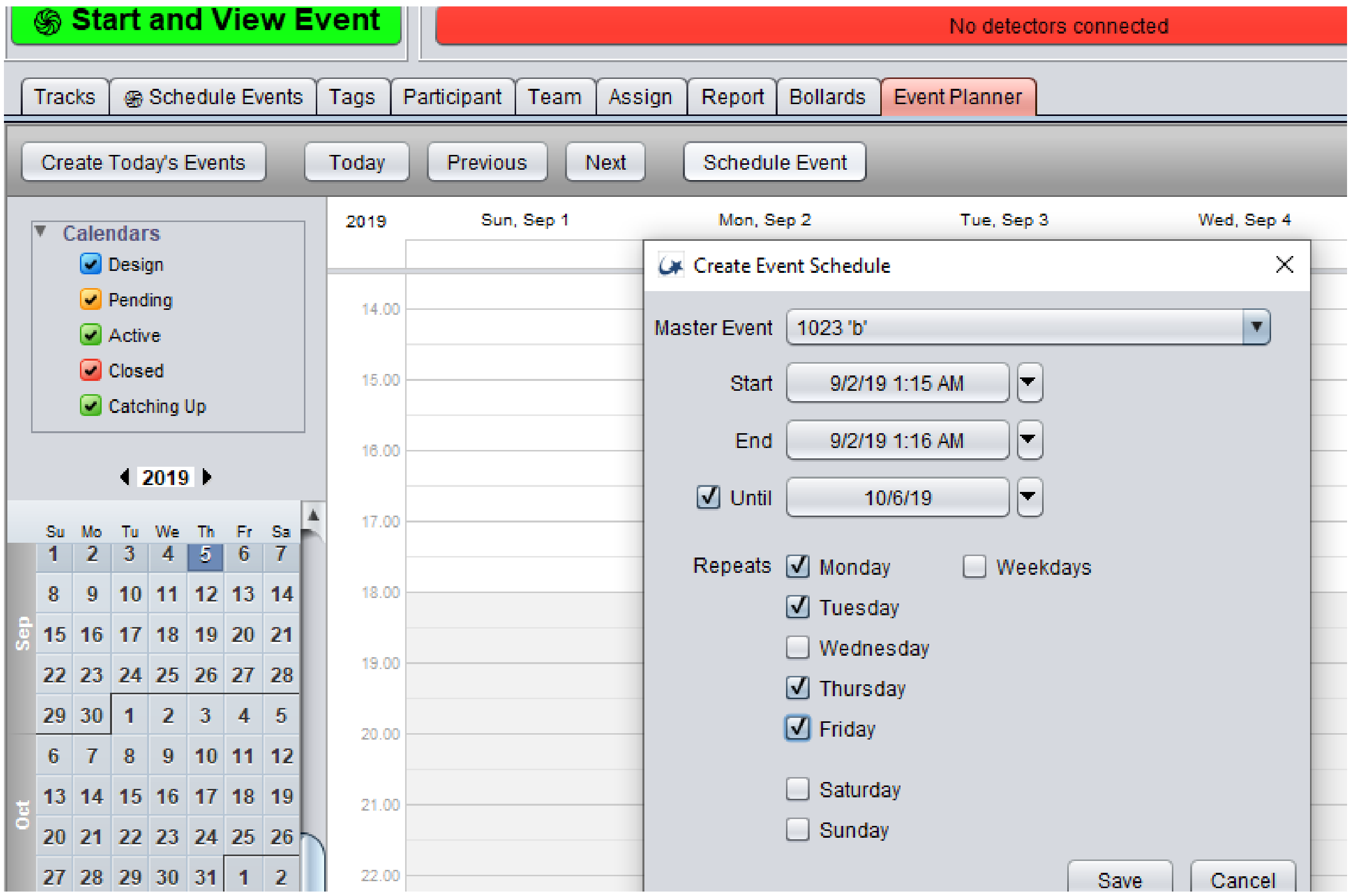
On the screen above , to the right you will see the “Tag Detections” page. This will show the tag detects in real time. Note that the tags have enter and leaving. This is because Orbiter positions on tags and thus allows first tag seen, last tag seen, and corkscrew detections.   All tags seen on the right side of the page have been entered into the data base.

Looking closely at Tag Detections you will see an 8hex number. It looks something like this 00000001 or A123UKL8. If you ever want to find the secrete integrated circuit number that has been written onto the chip inside your RFID, just scan the tag and see it here.

Tip: Think of Tag Detections as a chalk board that can be erased without harm. To read chip identifications it is easier to click inside the Tag Detection area and hold Ctrl button and A, to highlight them all in Blue. Then click Delete button. This clears the screen so when you pass a tag, you can identify which tag you scanned clearly.

At the bottom of the “Assign” screen is a “white area”. This is where detailed messages about your reader status is shown. If there is a connectivity problem, you will likely find the reason here.

C.2.1 Event Planner: The PE Class Scheduler with automated start and stop



No detectors connected

Calendars
- ✓ Design
- ✓ Pending
- ✓ Active
- ✓ Closed
- ✓ Catching Up

Create Event Schedule

Master Event: 1023 'b'

Start: 9/2/19 1:15 AM

End: 9/2/19 1:16 AM

✓ Until: 10/6/19

Repeats:
- ✓ Monday
- ✓ Tuesday
- ☐ Wednesday
- ✓ Thursday
- ✓ Friday
- ☐ Saturday
- ☐ Sunday
- ☐ Weekdays

This ambitious program was made for schools. Orbiter is the only RFID software program able to automatically schedule classes, stop and start new classes. It allows a school to schedule as many teachers, class period and students to use one Orbiter reader simultaneously.

The purpose is to free the teacher from having to interact with the Orbiter computer and concentrate on the kids. This is because once this is set up, then the teacher can collect reports.

To set the program up, go to “Schedule Event Tab” and create an event for each PE Period. Be sure to set the “Start” type to “As Scheduled” and NOT on “Button Click” The set the

"Duration" to the number of minutes in the class. Normally this is 40 minutes.

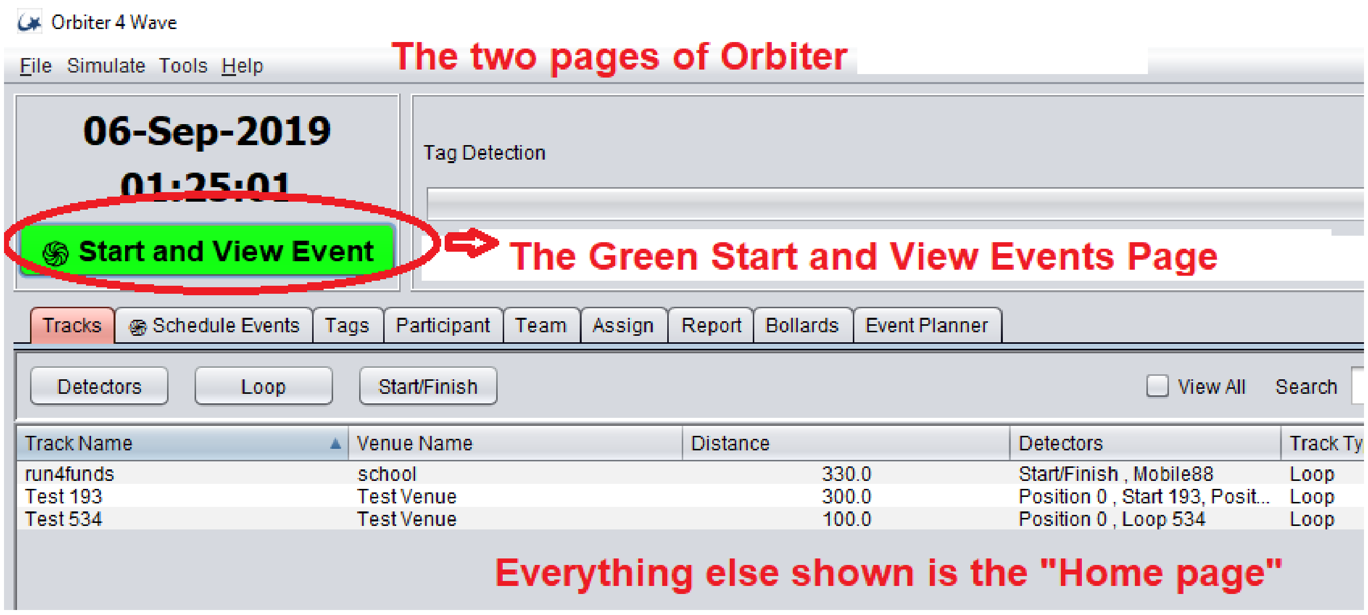
Use your All-In-One to load your kids into your class.

Next go to the event planner and select the event and then set the times and days for the event. This is seen on the "Create Event Schedule". Put an end date on "Until" click the days of the week and "Save". Your class periods for the entire year are now made. You will see them on the calendar replicated.

One this is set up at the beginning of each semester, then the machine automatically turns on and off and keeps track of the kids.

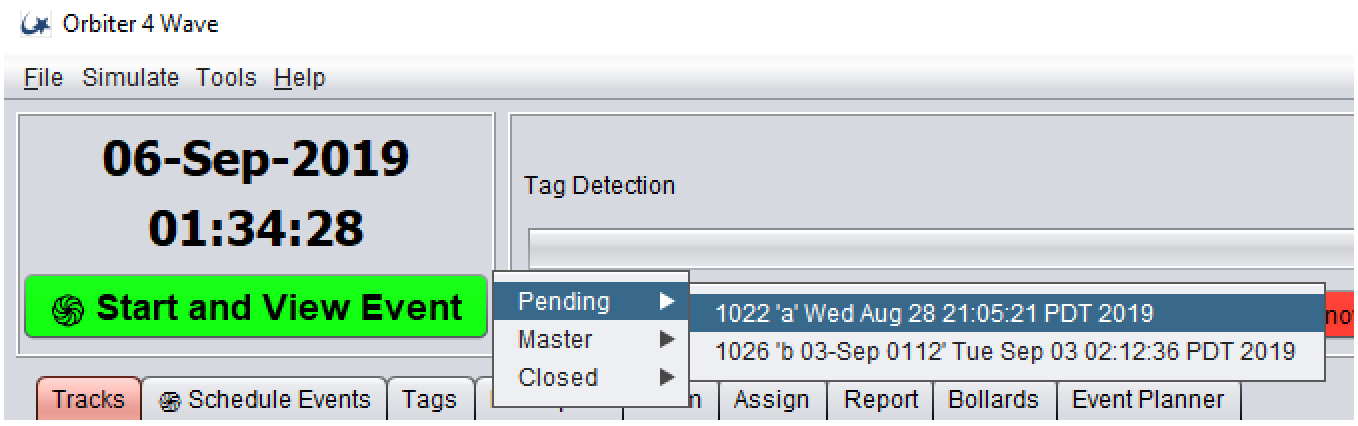
The machine can also trade tags between students and classes. However, we have found this is more difficult for teachers to administer and most like having every student have their own tags.

<u>G . 1 : The Green Start and View Events Page. The Second page of importance</u>.

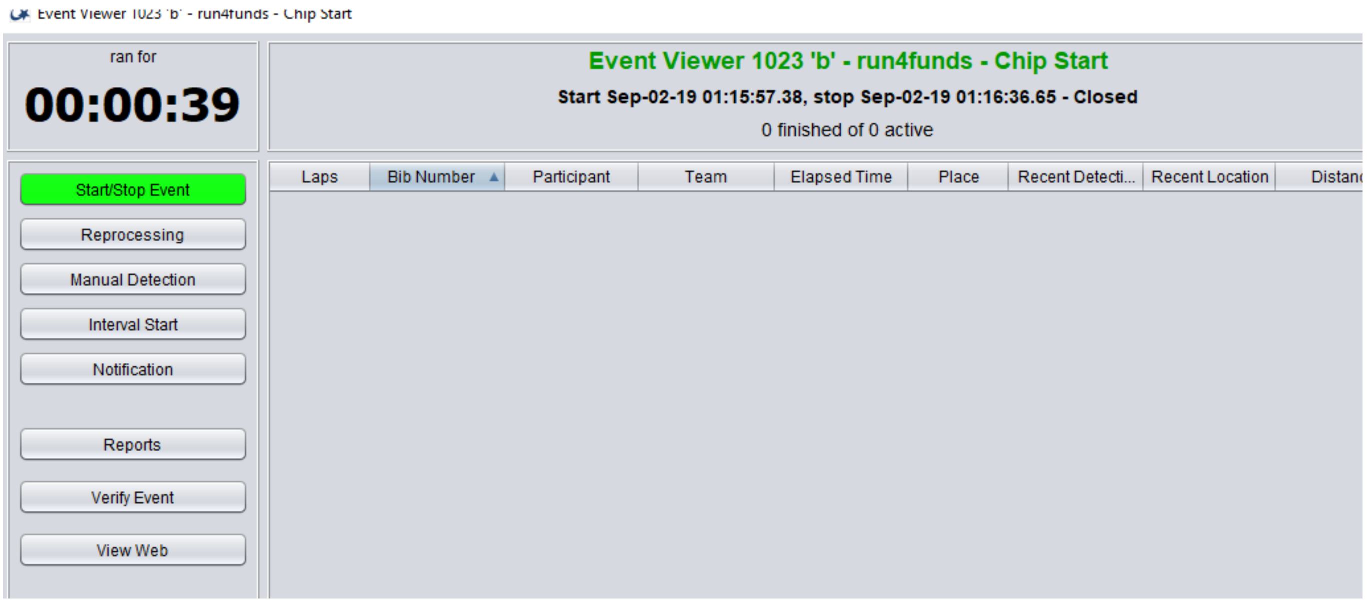


Track Name	Venue Name	Distance	Detectors	Track Ty
run4funds	school	330.0	Start/Finish , Mobile88	Loop
Test 193	Test Venue	300.0	Position 0 , Start 193, Posit...	Loop
Test 534	Test Venue	100.0	Position 0 , Loop 534	Loop

Click the green button and then choose an event.



The real time “Event Viewer” then is shown.



Event Viewer 1023 'b' - run4funds - Chip Start

ran for

00:00:39

Event Viewer 1023 'b' - run4funds - Chip Start

Start Sep-02-19 01:15:57.38, stop Sep-02-19 01:16:36.65 - Closed

0 finished of 0 active

Laps	Bib Number ▲	Participant	Team	Elapsed Time	Place	Recent Detecti...	Recent Location	Distan

On the left side of the event viewer are controls. “Start / Stop Events” is where you “Start” your event. “Reprocessing” is where you confirm a change setting in the middle or after a race. It also is used to reset the start time if needed. Manual Detection is where you insert a tag detect in event one is missed. Interval Start is how Intervals are started. This button will change to “Wave Start” when waves are required. Waves are often used in Triathlons. Reports are where Standard and Custom reports are gained. Verify Event is where you can double check your set up of an event.

The main grey area which is blank above, is where your live real time detections with lap counts and time is shown. In the Columns “Elapsed Time” is the race time for each participant. Recent Detection time is the “Gun Start” Time for each participant. It is from the time the start button is clicked to the time the detection was gained. By clicking this column with a down arrow, as people pass the finish they will be sorted to the top. This is also called “announcer” mode.

On the bottom right is “O”, “-“ and “+” buttons. Clicking “O” hides the left column from view. This is done if the screen is connected to a display with HDMI so participants may view results on a remote LED. “-“ makes the type set smaller. “+” makes the type set larger.

The columns may be sorted by clicking the column header and sliding them right or left. They also may be minimized to hide unwanted columns from view.